

Web 编程实践教程

丁振凡 主 编
赵 硕 副主编

超值配套赠送

- PPT电子课件
- 网络教学平台本机试用版
- 课程教学资源库

清华大学出版社

华东交通大学教材建设基金资助项目

Web 编程实践教程

丁振凡　主编
赵　硕　副主编

清华大学出版社
北　京

内 容 简 介

本书结合网络教学平台的应用开发实践，较为系统地介绍了Web程序开发的主要内容。全书共11章，内容包括ASP编程基础、HTML语言介绍、VBScript介绍、ASP的内置对象、ASP访问数据库、JavaScript脚本语言、层叠样式表CSS、DHTML编程、XML技术与应用、AJAX技术、网络教学综合应用实例。本书在内容讲述上由浅入深，注重理论与实际的结合；书中例题精炼，融知识性和实用性于一体；每章均配有丰富的习题和教学课件。

本书的突出特点是将Web编程中的客户端技术与服务器端技术进行了系统的融合整理，有利于培养学生综合分析问题和解决问题的能力。

本书既可作为高等院校Web程序设计和Web编程技术的教材，同时也可作为广大自学者和软件开发人员的参考用书。

图书在版编目（CIP）数据

Web编程实践教程/丁振凡主编．—北京：清华大学出版社，2011.1

ISBN 978-7-302-24256-7

I. ①W…　II. ①丁…　III. ①主页制作-程序设计-教材　IV. ①TP393.092

中国版本图书馆CIP数据核字（2010）第249514号

责任编辑：杜长清
封面设计：刘　超
版式设计：魏　远
责任校对：王国星
责任印制：
出版发行：清华大学出版社　　**地　址：**北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　编：**100084
社 总 机：010-62770175　　**邮　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185×260　**印　　张：**17.5　**字　　数：**404千字
版　　次：2011年1月第1版　**印　　次：**2011年1月第1次印刷
印　　数：1～4000
定　　价：33.00元

产品编号：040684-01

前　　言

随着 Internet 应用的普及，社会对 Web 应用程序开发人员的需求越来越多。为了适应信息技术的发展和社会的需求，不少高校开始在计算机及相关专业开设 Web 程序设计方面的课程。根据教学实际需要，再结合近年来在 Web 应用开发和 Web 教学中的经验，笔者编写了本教材。

Web 应用开发的基础是网页，在此基础上结合 Web 服务器端的数据访问技术以及客户浏览器端的动态交互技术实现 Web 应用的动态交互性。在服务器端技术中使用比较普遍的有 ASP、JSP、PHP 等技术。考虑到微软 ASP 应用开发环境具有简单易学的特点，因此，本书选择用 ASP 技术作为服务器端的开发环境进行介绍，客户浏览器端的动态交互处理则选用广泛采用的 JavaScript 脚本语言进行介绍。另外，由于 XML 技术在客户端和服务器端的应用开发中都得到了广泛使用，因此，本书对 XML 技术及 AJAX 技术的应用也进行了重点介绍。

全书共 11 章，第 1 章介绍 ASP 编程基础，重点是让读者熟悉 Web 环境的搭建，了解 ASP 程序的基本特点；第 2 章介绍 HTML 语言，让读者了解常用 HTML 标记的使用；第 3 章介绍 VBScript 语言，它是服务端推荐采用的脚本语言；第 4 章介绍 ASP 内置对象的使用；第 5 章介绍 ASP 访问数据库；第 6 章介绍 JavaScript 脚本语言；第 7 章介绍层叠样式表 CSS；第 8 章介绍 DHTML 编程；第 9 章简要介绍 XML 的技术与应用；第 10 章介绍 AJAX 技术；第 11 章给出了网络教学的几个综合应用实例，以培养读者对 Web 程序设计技术的综合应用能力。本课程是一门实践性较强的课程，不仅要求学生掌握基本理论、基本技术和基本方法，更重要的是使学生具有较强的实际操作应用能力。课程的实验部分安排了丰富的内容供教师选用，并且每章的后面均配有习题。

本书是作者多年来教学和软件开发经验的总结。作者对书中内容进行了精心的设计和安排，按照由浅入深、循序渐进的原则进行组织，力求实现内容丰富、结构清晰。书中程序样例大多简短实用，易于教师教学使用和读者学习；书中所有代码均经过调试，大部分案例来源于网络教学平台的开发实践，具有较大的实际应用价值。Web 编程的一个关键点是能很好地选择和使用技术，本书在对 Web 编程的客户端和服务器端的理论与技术进行归纳整理的同时，注意技术的融合与运用，使读者在一个渐进的学习过程中把握这些技术的特点，并应用于实际项目的开发中。

本书不仅适合教学，也适合使用 Web 应用开发的用户学习和参考。阅读本书，并结合上机实训进行练习，就能在较短的时间内基本掌握 Web 应用开发的基本技术。另外，本书为教师配有教学课件，并提供配有实例的源程序，需要的教师可登录华东交通大学的网络教学网站（http://cai.ecjtu.jx.cn/）下载。

本书第 1、2 章由齐齐哈尔大学的赵硕老师编写，第 3～11 章由华东交通大学丁振凡教

授编写。感谢华东交通大学的蔡体健、李卓群、莫佳、王鹏鸣等老师在本书编写过程中提出了不少宝贵意见。感谢研究生吴根斌仔细阅读了本书，并对书中习题进行了解答。由于编者水平所限，疏漏和错误之处在所难免，恳请读者批评指正。

编　者

2010 年 10 月于南昌

目　　录

第 1 章　ASP 编程基础 1
1.1　Web 基础知识 1
1.1.1　Web 工作原理 1
1.1.2　Web 页与 Web 站点 2
1.2　ASP 简介 2
1.3　IIS 的安装与配置 3
1.3.1　安装 IIS 3
1.3.2　启动和停止 IIS 3
1.3.3　配置 IIS 3
1.4　ASP 程序初步介绍 5
1.4.1　简单示例 5
1.4.2　ASP 脚本语言设定 6
1.4.3　服务器端包含的文件 7
本章小结 8
习题 8
第 2 章　HTML 语言介绍 10
2.1　HTML 概述 10
2.1.1　HTML 文档结构 10
2.1.2　常用 HTML 编辑工具 11
2.2　HTML 文本设计 12
2.2.1　设置 body 属性 12
2.2.2　段落格式化 13
2.2.3　字符格式化 14
2.2.4　使用列表格式 16
2.3　使用表格 18
2.3.1　创建基本表格 18
2.3.2　表格设置 18
2.4　在网页中加入多媒体 21
2.4.1　使用图像 21
2.4.2　使用字幕和背景音乐 23
2.5　使用框架 24
2.5.1　框架网页的基本结构 24

2.5.2　框架的设置 25
2.6　使用超链接 26
2.6.1　理解超链接和路径 26
2.6.2　创建文件链接 27
2.6.3　创建锚点链接 27
2.6.4　创建邮件链接 28
2.7　使用表单 28
2.7.1　表单处理概述 28
2.7.2　INPUT 标记型表单控件的使用 29
2.7.3　其他表单控件 31
本章小结 33
习题 33
第 3 章　VBScript 介绍 35
3.1　VBScript 概述 35
3.2　VBScript 的数据表示 36
3.2.1　VBScript 的数据类型 36
3.2.2　VBScript 的常量、变量与数组变量 36
3.2.3　VBScript 运算符 38
3.3　VBScript 的流程控制语句 40
3.3.1　if 语句 40
3.3.2　Select Case 语句 41
3.3.3　循环语句 42
3.4　VBScript 的过程定义与调用 44
3.4.1　Sub 过程及其调用 45
3.4.2　Function 过程及其调用 45
3.5　VBScript 中的内部函数 46
3.5.1　转换函数 46
3.5.2　字符串函数 46
3.5.3　日期和时间函数 47
3.5.4　数学函数 47
3.5.5　检验函数 49
3.5.6　输入与输出函数 49
本章小结 50
习题 51
第 4 章　ASP 的内置对象 54
4.1　Request 对象 54
4.1.1　Form 集合 55

4.1.2 QueryString 集合 58
4.1.3 Cookies 集合 59
4.1.4 ServerVariables 集合......60
4.2 Response 对象......62
4.2.1 Response 对象的属性......62
4.2.2 Response 对象的方法......63
4.2.3 Response 对象的数据集合......65
4.3 Session 对象......67
4.3.1 Session 对象的属性......67
4.3.2 Session 对象的方法......68
4.3.3 Session 对象的事件......68
4.4 Application 对象......69
4.4.1 Application 对象的方法......70
4.4.2 Application 对象的事件......70
4.4.3 Global.asa 文件......71
4.5 Server 对象......75
4.5.1 Server 对象的属性......75
4.5.2 Server 对象的方法......75
本章小结......77
习题......77
第 5 章 ASP 访问数据库......80
5.1 结构化查询语言 SQL......80
5.1.1 SQL 命令的基本构成......80
5.1.2 SQL 查询......81
5.1.3 其他 SQL 语句......85
5.2 ADO 对象模型简介......87
5.2.1 ADO 内幕......87
5.2.2 ADO 对象和数据集合......88
5.3 用 Connection 对象连接数据库......88
5.3.1 Connection 对象的常用属性和方法......89
5.3.2 连接数据库......90
5.3.3 用 Connection 对象执行 SQL 语句......92
5.3.4 Connection 对象的数据集合......94
5.3.5 Connection 对象的事务处理......95
5.4 用 Recordset 对象访问数据库......97
5.4.1 Recordset 对象的创建......97
5.4.2 记录集游标及移动方法......98

5.4.3 访问记录的数据内容 99
5.4.4 记录集的分页显示 100
5.4.5 记录的添加与编辑修改 103
5.5 Command 对象 111
5.5.1 Command 对象的常用属性 112
5.5.2 Command 对象的常用方法 112
5.5.3 Command 对象的数据集合 113
5.5.4 通过 Command 对象调用存储过程 114
本章小结 115
习题 116
第 6 章 JavaScript 脚本语言 120
6.1 JavaScript 的基本语法成分 120
6.1.1 在网页中插入 JavaScript 代码 120
6.1.2 数据类型与变量 121
6.1.3 JavaScript 运算符 122
6.1.4 内置函数 123
6.2 程序流程控制语句 124
6.2.1 条件语句 124
6.2.2 循环语句 125
6.3 内置对象 126
6.3.1 String 对象 126
6.3.2 Array 对象 128
6.3.3 Date 对象 129
6.3.4 Math 对象 129
6.4 自定义函数 130
6.4.1 函数的定义 130
6.4.2 函数的调用 131
6.5 用户自定义对象 131
6.5.1 自定义对象创建方式 132
6.5.2 JavaScript 对象的操作 133
6.5.3 定义对象属性 134
6.5.4 定义对象方法 135
本章小结 137
习题 137
第 7 章 层叠样式表 CSS 140
7.1 样式表的定义与引用 140
7.2 样式表的种类 142

7.3　CSS 属性..145
7.3.1　字体属性..145
7.3.2　文本属性..146
7.3.3　颜色和背景属性..147
7.3.4　列表属性..148
7.3.5　边框、边距和间隙属性..149
7.3.6　定位与布局属性..151
本章小结...155
习题...155
第 8 章　DHTML 编程..159
8.1　浏览器对象模型...159
8.1.1　window 对象..159
8.1.2　document 对象...164
8.1.3　location 对象...168
8.1.4　history 对象...168
8.1.5　external 对象...171
8.1.6　navigator 对象..171
8.1.7　screen 对象..171
8.2　JavaScript 的事件处理...171
8.2.1　JavaScript 事件处理方法..172
8.2.2　常见事件一览...173
8.2.3　document 的常用事件...174
8.2.4　表单处理的常用事件..178
8.2.5　表单中的控件...181
本章小结...182
习题...183
第 9 章　XML 技术与应用...186
9.1　XML 文档格式...186
9.2　XML 文档对象模型...188
9.2.1　DOMDocument 对象..189
9.2.2　XMLDOMNode 对象..190
9.2.3　XMLDOMNodeList 对象..193
9.3　XML 文档的显示处理..193
9.3.1　利用 CSS 显示..193
9.3.2　使用 Xpath 查找结点..195
9.3.3　利用 XSL 实现显示..199
9.4　在服务器端访问和处理 XML 文档..201

9.5　在客户端访问和处理 XML 文档 207
9.5.1　通过脚本装载和处理 XML 文档 207
9.5.2　XML 数据岛 209
本章小结 214
习题 215
第 10 章　AJAX 技术 217
10.1　什么是 AJAX 217
10.2　XMLHttpRequest 对象的使用 218
10.2.1　创建 XMLHttpRequest 对象 218
10.2.2　XMLHttpRequest 对象的属性 218
10.2.3　XMLHttpRequest 对象的方法 219
10.2.4　在 Web 服务器端使用 XMLHttpRequest 对象 221
10.3　AJAX 应用举例 221
10.3.1　样例 1——网络考试中避免并发交卷的处理 221
10.3.2　样例 2——作品的投票处理 223
10.3.3　样例 3——页面元素间的关联处理 226
10.4　在 AJAX 中使用 JSON 228
10.4.1　JSON 的具体形式 228
10.4.2　JSON 数据格式解析 229
本章小结 231
习题 231
第 11 章　网络教学综合应用实例 233
11.1　网上答疑子系统 233
11.1.1　数据库表格设计 233
11.1.2　辅助包含文件 233
11.1.3　学生端的设计 234
11.1.4　教师端的设计 237
11.2　基于 XML 的单元自测应用 243
11.2.1　功能概述 243
11.2.2　测试试卷的 XML 表示 243
11.2.3　考试解答界面的生成及显示处理 245
11.2.4　考试的解答记录、交卷评分及答案对比的显示 249
11.3　网络课件导航菜单的设计 252
11.3.1　导航菜单的设计要求 252
11.3.2　基于 XML 的导航菜单设计 253
11.4　网络考试系统 256
11.4.1　考试界面布局 256

11.4.2　考试组卷程序 257
11.4.3　试题显示处理程序 259
11.4.4　答题卡显示处理程序 260
11.4.5　交卷评分显示处理程序 261
本章小结 262
习题 263

参考文献 265

第 1 章　ASP 编程基础

1.1　Web 基础知识

1.1.1　Web 工作原理

Web 全称为 World Wide Web，缩写为 WWW。Web 有许多译名，如环球网、万维网、全球信息网等。目前尚无对 Web 形成一致的定义，简单地说，Web 是 Internet 提供的一种服务，通过它可以访问分布于 Internet 主机上的信息资源，它是存储在全世界 Internet 计算机中、数量巨大的链接文档的集合。

Web 以客户机/服务器模式运行。信息资源以页面形式存储在 Web 服务器上，用户通过客户端的 Web 浏览器向 Web 服务器发出查询请求；Web 服务器根据客户端请求的内容做出响应，并将存储在服务器上的某个页面发送给客户端；Web 浏览器对收到的页面进行解释并将页面显示给用户。浏览器与 Web 服务器间的信息传输采用超文本传输（HTTP）协议，如图 1-1 所示。

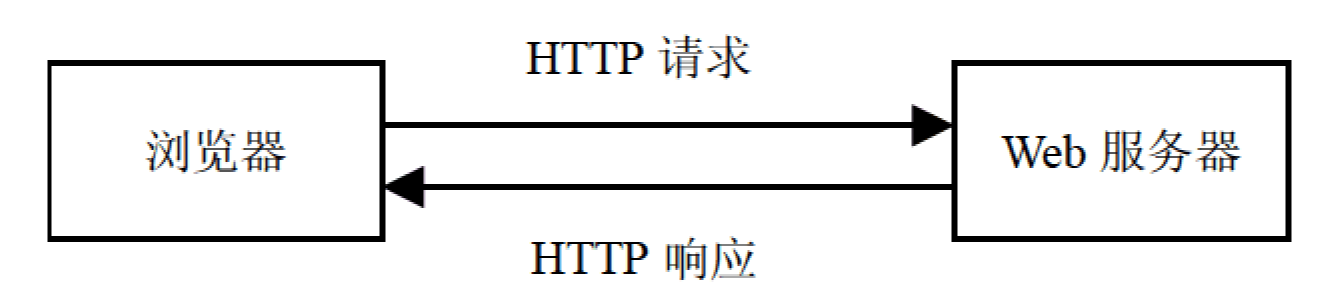


图 1-1　Web 的工作原理

Web 服务器通常是指安装了服务器软件的计算机，常见的 Web 服务器软件包括 Microsoft Internet Information Server（IIS）、Microsoft Personnal Web Server（PWS）和 Apache HTTP Server 等，常用的 Web 浏览器软件有 Netscape Navigator（NN）、Microsoft Internet Explorer（IE）和 Mozilla Firefox（火狐）等。

URL 即统一资源定位符，是一种唯一地标识 Internet 上计算机、目录和文件的位置的命名规则。URL 用于指定获得 Internet 上资源的方式和位置，通常也称为 URL 地址、网站地址或网址，其一般形式如下：

<方式>://<主机名>:<端口>/<目录>/…/<文件名>

其中：

- <方式>指定访问该资源所使用的 Internet 协议，常用形式有 http（超文本传输协议）、ftp（文件传输协议）、mailto（电子邮件地址）、news（网络新闻组）、telnet（远程登录服务）和 file（本地文件）等。
- <主机名>指定 Web 服务器的 IP 地址或域名地址。IP 地址是唯一标识网络上某一

主机的地址，它将计算机标识为一个 32 位地址，可以用带英文句点的十进制数来表示。域名地址也称为 DNS 地址，由 4 个部分组成，常用形式为“机器名.单位名.单位类别.国别”。例如，华东交通大学的 WWW 服务器地址为 www.ecjtu.jx.cn。

【注意】localhost 是一个特殊的名称，它代表本机地址。

- <端口>指定 Web 服务器在该主机上所使用的 TCP 端口，默认端口是 80。该端口通常不需要指定，只有当 Web 服务器不使用默认端口时才需要指定。
- <目录>是 Web 服务器上信息资源相对于 Web 服务器的根目录或虚拟目录所处的目录路径，每一级目录以正斜杠（/）隔开。
- <文件名>由基本文件名和扩展名两部分组成，如 index.htm。

以下是 URL 的一些例子：

http://www.sina.com.cn/

http://cai.ecjtu.jx.cn/java/default.htm

ftp://ftp.w3.org/pub/www/doc

1.1.2　Web 页与 Web 站点

Web 页通常称为网页，它一般由 HTML 文件组成，其中包含相关的文本、图像、声音、动画、视频及脚本命令等，位于特定计算机的特定目录中，其位置可以根据 URL 确定。按照 Web 服务器响应方式的不同，可以将 Web 页分为静态网页和动态网页。

Web 站点提供 Web 服务访问的地址。一般的 Web 站点由一组相关的 HTML 文件和其他文件组成，这些文件存储在 Web 服务器上。当用户访问一个 Web 站点时，该站点中有一个页面总是被首先打开，该页面称为首页或主页，如 index.html。

1.2　ASP 简介

ASP 全称为 Active Server Pages，是一种由 Microsoft 公司开发的服务器端脚本语言运行环境，它可以结合 HTML 语言和 ActiveX 组件建立动态、交互、高效的 Web 服务器端应用程序。当一个用户浏览器从 Web 服务器请求一个 ASP 网页时，Web 服务器会将这个 ASP 文件发送给 Web 服务器的 ASP 引擎，ASP 引擎将该 ASP 网页中所有的服务器端脚本（<%和%>之间的代码）进行处理，并将输出结果转换成 HTML 代码，然后将处理后的完整 HTML 代码发送给用户浏览器。ASP 的工作原理如图 1-2 所示。在 ASP 程序中可以通过 ActiveX Data Object 对象实现对数据库的访问处理。

从软件的技术层面看，ASP 具有如下特点：

（1）无须编译。ASP 脚本与 HTML 集成一体，可直接由 ASP 引擎解释执行。

（2）易于生成。使用常规文本编辑器即可进行 ASP 程序代码的编写，同时也可以使用 FrontPage 和 Dreamweaver 等网页制作工具。

（3）独立于浏览器。ASP 脚本在站点服务器端执行，用户端的浏览器不需要支持它。

（4）面向对象。在 ASP 脚本中可以方便地引用系统组件和 ASP 的内置组件，另外还能通过定制 ActiveX 服务器组件来扩充功能。

（5）与任何 ActiveX Scripting 语言兼容。除了可使用 VBScript 和 JScript 语言进行设计外，还可通过 Plug-in 的方式使用由第三方所提供的其他 Scripting 语言。

（6）源程序码不会外漏。由于 ASP 在服务器端解释执行，在客户端的浏览器上看到的是执行后的结果，所以开发者无须担心他人下载程序代码，从而增加了网站的安全性。



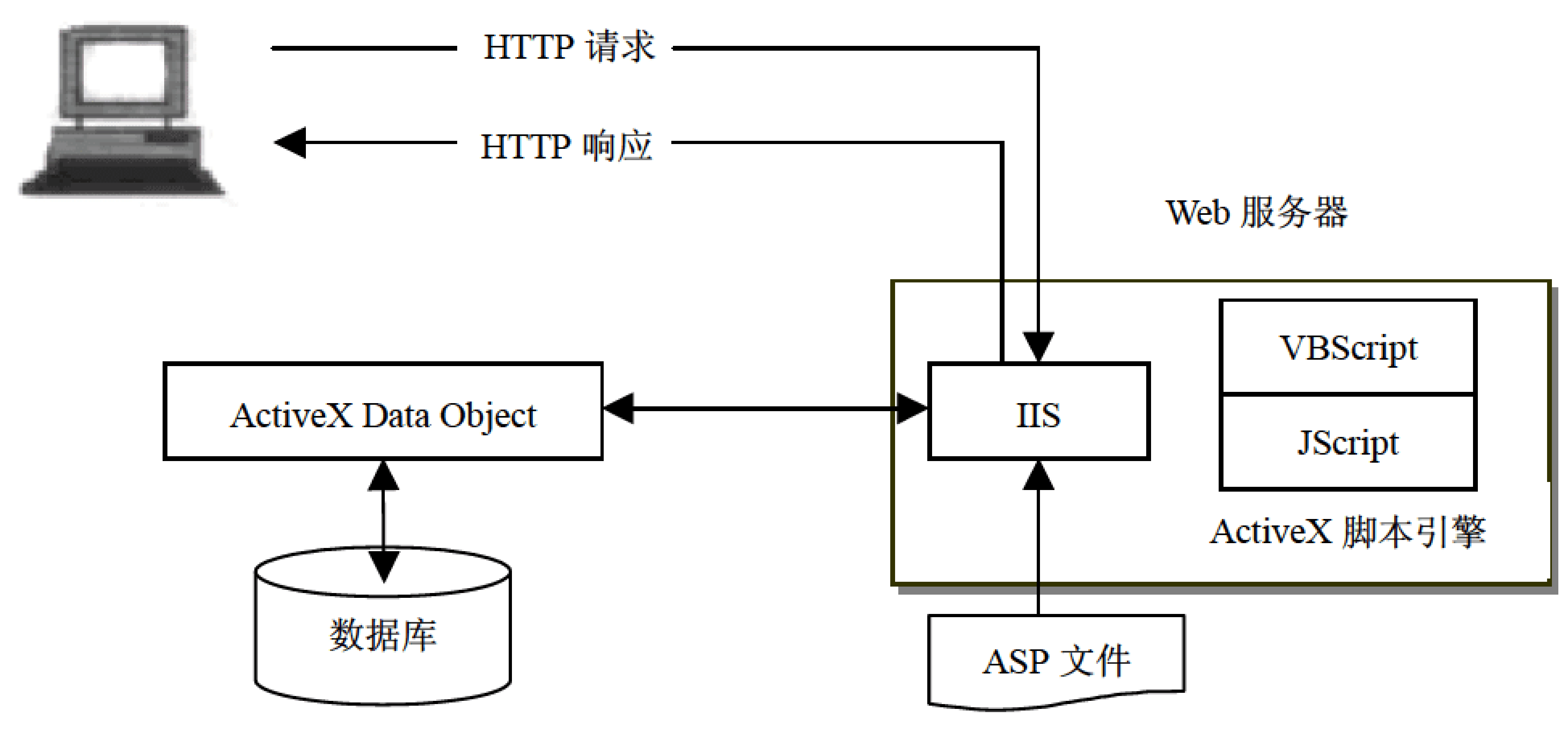


图 1-2　ASP 的工作原理

1.3　IIS 的安装与配置

1.3.1　安装 IIS

若操作系统中还未安装 IIS 服务器，可打开控制面板，然后单击“添加/删除程序”，在弹出的对话框中选择“添加/删除 Windows 组件”选项，在 Windows 组件向导对话框中选中“Internet 信息服务（IIS）”复选框，然后单击“下一步”按钮，按向导指示，即可完成对 IIS 的安装。

1.3.2　启动和停止 IIS

Internet 信息服务简称为 IIS，选择 Windows 的“开始”→“所有程序”→“管理工具”→“Internet 信息服务（IIS）管理器”命令，在弹出的对话框中单击▶图标即可启动“Internet 信息服务”，如图 1-3 所示。在服务器已启动的情况下，单击■图标，则停止 IIS 服务器。

1.3.3　配置 IIS

用鼠标右键单击“默认 Web 站点”，在弹出的快捷菜单中选择“属性”命令，此时即可

打开“默认网站 属性”对话框，如图 1-4 所示。在该对话框中可完成对站点的全部配置。



图 1-3　Internet 信息服务（IIS）管理器



图 1-4　“默认网站 属性”对话框

1．配置主目录

当 IIS 安装后，系统自动创建了一个默认的 Web 站点，该站点的主目录默认为 C:\Inetpub\wwwroot。单击“主目录”标签，切换到“主目录”选项卡，如图 1-5 所示，在该选项卡中可实现对主目录的更改或设置。

2．设置主页文档

单击“文档”标签，可切换到“文档”选项卡，如图 1-6 所示。主页文档是在浏览器地址栏中只输入网站域名，未指定要访问的网页文件时，Web 服务器默认提供的页面文件。IIS 默认的主页文档只有 default.htm 和 default.asp，根据需要用“添加”和“删除”按钮，可为站点设置所能解析的主页文档。



图 1-5　默认网站的主目录设置



图 1-6　默认网站的文档设置

3．建立虚拟目录

如果应用存储在 Web 服务器的磁盘上任意一个路径下，要进行 Web 发布，可以建立一个虚拟目录，每个虚拟目录有一个别名，用户通过浏览器访问只需在访问相应站点的主机名和目录路径之间再通过斜杠符隔开添加指定别名，Web 服务器就会根据虚拟目录定义

的实际存储路径提取要访问的资源文件，如图 1-7 所示。

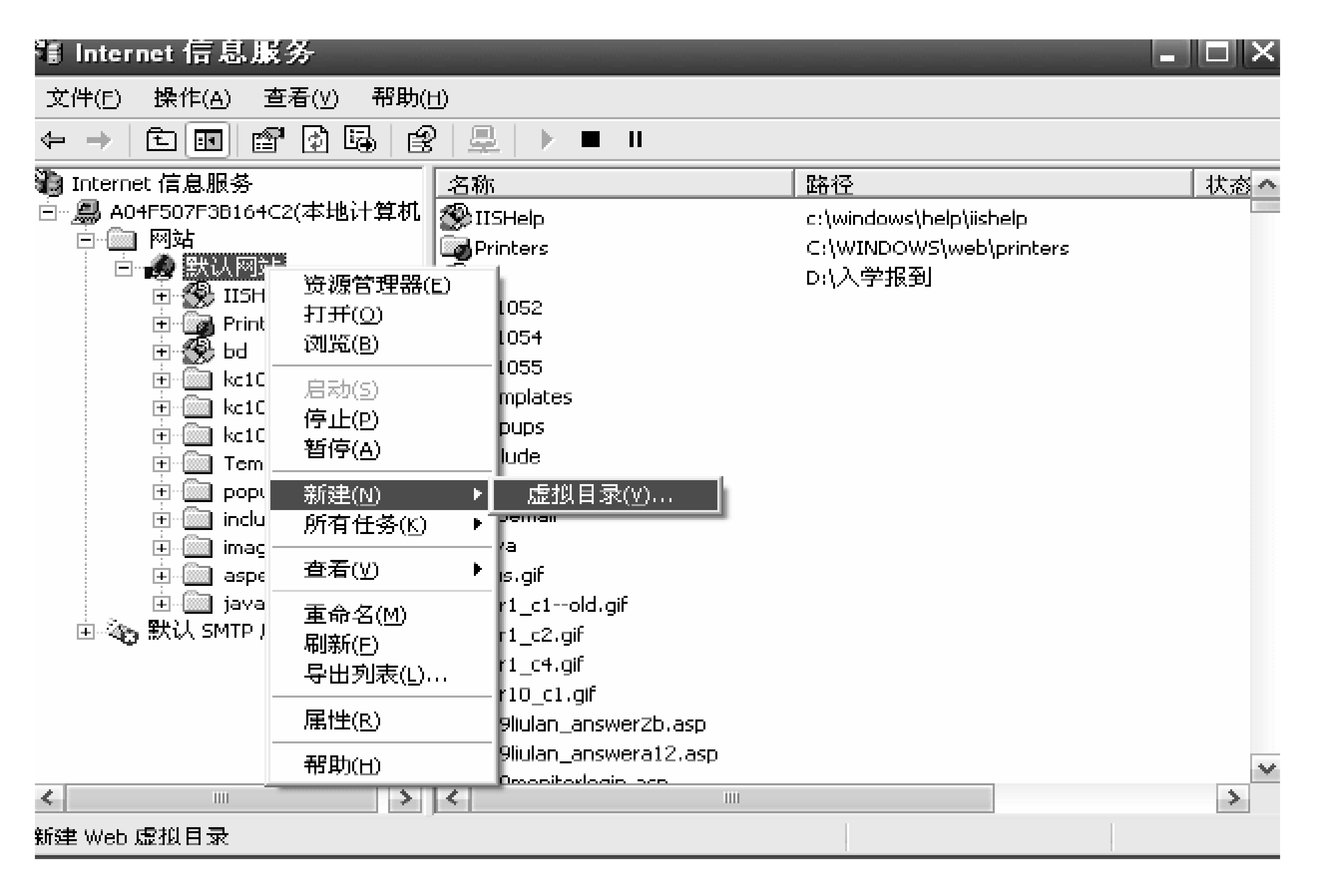
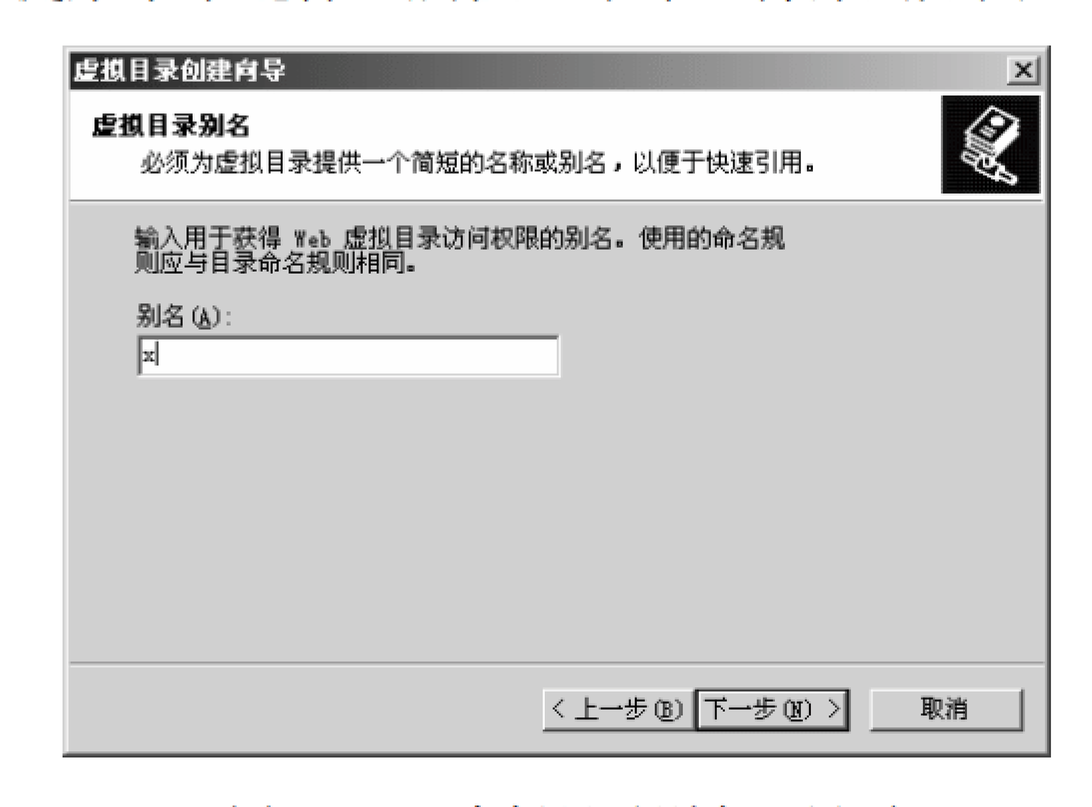


图 1-7　在 IIS 中新建虚拟目录

在默认网站上单击鼠标右键，从弹出的快捷菜单中选择“新建”→“虚拟目录”命令，将弹出“虚拟目录创建向导”对话框，单击“下一步”按钮，出现“虚拟目录别名”界面，如图 1-8 所示。在输入别名后单击“下一步”按钮，将出现路径选择对话框，依次按照提示操作即可完成虚拟目录的创建设置。

要修改虚拟目录的属性，可在 IIS 控制台中选中虚拟目录后单击鼠标右键，在弹出的快捷菜单中选择“属性”命令，将弹出如图 1-9 所示的对话框，根据需要进行设定。



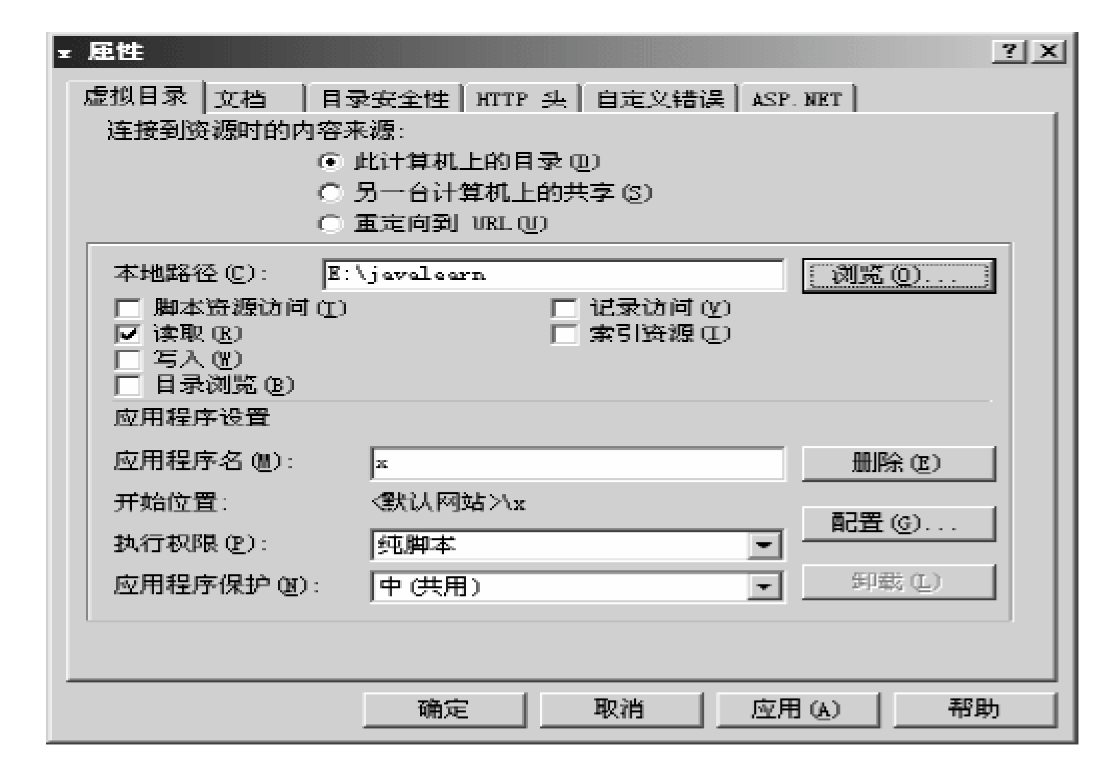
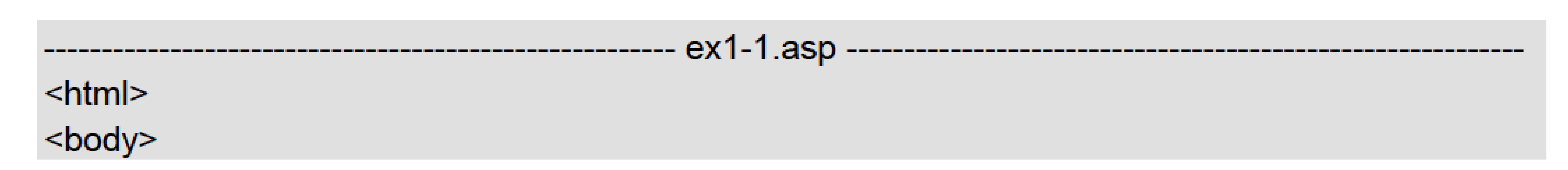


图 1-8　“虚拟目录别名”界面　　　　图 1-9　虚拟目录属性

1.4　ASP 程序初步介绍

1.4.1　简单示例

【例 1-1】　简单 ASP 程序



```
---------------------------------------------------- ex1-1.asp ----------------------------------------------------
<html>
<body>
```

```
<center><b><font color="green" size="6">第一个 asp 程序</font></b>
</center><BR>
<%For i=3 To 7 %>
<FONT SIZE=<% = i %>>  本行字体大小是<%=i%>号字! </ FONT><BR>
<%Next %>
</body>
</html>
```

运行结果如图 1-10 所示。



图 1-10　ex1-1.asp 程序访问结果

【说明】可以看出，ASP 文件中脚本代码和 HTML 文本混合在一起，随着脚本代码中循环的执行，循环内所包含的 HTML 文本将重复出现。脚本代码中的<% = i %>表示在页面中输出 i 的值，脚本在页面中哪个位置，则输出也将嵌入在页面的该位置。

1.4.2　ASP 脚本语言设定

ASP 的默认脚本语言是 VBScript，它也是编写 ASP 最合适的脚本语言。除了 VBScript 之外，如果需要使用其他脚本语言编写 ASP 网页，则应在文件的开头位置使用<%@ language=…%>加以设定，注意@和 language 之间一定要有空格。

以下 ASP 文件所用的脚本语言是 JavaScript。

```
<%@ language="javascript" %>
<html>
<body>
<%
var d = new Date()
Response.Write(d)
%>
</body>
</html>
```

【说明】此时脚本中用 Response.Write 方法在页面上输出信息，与前面例子中<%=…%>

方式的作用相同。

【注意】VBScript 是不区分大小写的，但 JavaScript 是区分大小写的。例如，在 VBScript 中用小写 response.write，脚本也将正常执行，但在 JavaScript 中则一定要写成 Response. Write，否则会出错。

另外，还可以使用 SCRIPT 标记的 LANGUAGE 属性来设置所用的脚本语言，并用 RUNAT 属性指明脚本是在服务器端运行的。常用该形式定义服务器端要执行的函数。例如：

```
<SCRIPT LANGUAGE = "VBScript" RUNAT = "Server">
    function pickst(table,amount,knowledges,diff)
        ……
    end function
</SCRIPT>
```

【注意】在 SCRIPT 标记中如果未指定 RUNAT = "Server"属性，则表明其中的脚本代码是在客户端使用的。

1.4.3　服务器端包含的文件

如果一段代码要在多个 ASP 文件中重复使用，可以将该代码段安排在一个专门的文件中，用到该段代码的 ASP 文件可通过#include 命令将其内容插入到文档中。其语法格式如下：

<!-- #include file|virtual = Filename -->

其中，file 和 virtual 参数指定包含文件的路径类型。

- file 指定路径类型为使用#include 命令的文件所在文件夹的相对路径，被包含文件可以位于相同文件夹或子文件夹中，但它不能处于父文件夹中。
- virtual 指定路径类型为 Web 站点上虚拟目录的完整虚拟路径。
- Filename 参数指定要包含的文件名，必须包含文件扩展名，而且必须用引号将文件名括起来。被包含文件的扩展名可以任意命名，但建议使用.inc 扩展名。例如：

```
<!-- #include file = "adcvbs.inc" -->
<!-- #include virtual = "/scripts/tools/global.inc" -->
<!-- #include file = "md5.asp" -->
```

【注意】

（1）不能使用 ASP 程序代码动态设定“被包括”的#include 文件。例如：

```
<% name= test & ".inc"%>
<!--#include file="<%=name%>"-->
```

（2）使用#include 可以提高代码的重用性，达到“一处定义，多处使用”的目的。在实际应用中一般将常量定义、函数定义等安排在 inc 文件中，但从另一方面来说，#include 的使用也带来了效率上的下降，建议读者合理选择。

本 章 小 结

本章介绍了 ASP 的基础知识，包括 Web 访问的基本原理、IIS 服务器的安装与配置、ASP 的工作原理与特点以及 ASP 程序基本组成结构等。目的是让读者对 ASP 的运行环境和代码结构有一个基本的了解。

习　　题

1．选择题

（1）可以使用（　　）软件编辑 ASP 程序。

A．记事本　　B．FrontPage

C．Excel　　D．Dreamweaver

（2）以下说法正确的是（　　）。

A．只有微软公司推出的 Web 服务器才支持 ASP

B．从浏览器的菜单栏选择“查看”→“源文件”命令，即可看到 ASP 程序代码

C．在 ASP 程序中能使用 VBScript 语言，但不能使用 JavaScript 语言

D．浏览者只要从 IE 中选择查看源文件，即可看到 ASP 代码

（3）假设计算机的域名为 xxx.ecjtu.jx.cn，Web 主目录为 C:\Inetpub\wwwroot\，同时在此目录下还有一个 ASP 程序，其完整路径为 C:\Inetpub\wwwroot\test\ShowTime.asp。如果要在浏览器中执行此 ASP 程序，则必须在地址栏中输入的网址是（　　）。

A．http://xxx.ecjtu.jx.cn/ShowTime.asp

B．file://xxx.ecjtu.jx.cn/ShowTime.asp

C．http://Inetpub/wwwroot/xxx/ShowTime.asp

D．http://xxx.ecjtu.jx.cn/test/ShowTime.asp

（4）Web 服务器的概念主要是指（　　）。

A．提供 Web 服务的计算机

B．辅助信息处理的计算机

C．提供 Web 服务的程序

D．提供 Web 服务的公司

2．问答题

（1）简述 ASP 的工作原理与特点。

（2）ASP 的运行平台是什么？如何建立 Web 站点和虚拟目录？

3．操作题

（1）在 D 盘建立一个文件夹 test，用于存放编写的 ASP 程序，同时设置“默认网站”

的主目录为所建的文件夹。

（2）用记事本编写一个 ASP 程序，命名为 x.asp，存储到文件夹 test 下。用浏览器访问建立的文件（如 http://localhost/x.asp）。

```
---------------------------------------------------------- x.asp ----------------------------------------------------------
<html>
<body>
<font color="red" size="7">奥运纪念</font><BR>
<%For i=2 To 20 step 3%>
<span style="color=blue;font-size=<%=i%>pt;">2008 北京! </ span><BR>
<%Next %>
</body>
</html>
```

（3）将上述两个文件复制到另一个路径下，建立一个虚拟目录 you 指向文件所在的新路径。用浏览器访问文件（如 http://localhost/you/x.asp）。

（4）设置默认网站的“默认文档”为 x.asp，用浏览器访问网址 http://localhost/。对 x.asp 程序中的字体颜色、大小及文字等内容进行修改，观察网页内容的显示变化。另外，通过 IP 地址访问其他同学的网站。

第 2 章　HTML 语言介绍

2.1　HTML 概述

Web 信息发布是通过网页实现的，要编辑网页的内容首先要熟悉网页的基本语言——HTML。HTML 通过标记来指示要显示网页中的各个部分，即确定网页内容的格式。浏览器按照顺序阅读 HTML 文件，然后根据内容附近的 HTML 标记来解释和显示各种内容。需要注意的是，所有的标记都必须用尖括号（即小于号“<”和大于号“>”）括起来。

2.1.1　HTML 文档结构

HTML 文档的基本结构表示如下：

```
<HTML>
<HEAD>
<TITLE>标题文字</TITLE>
<HEAD>
<BODY>
<!--可插入文本、图像、动画、HTML 指令等-->
</BODY>
</HTML>
```

【说明】

（1）3 对顶级标记<HTML>、<HEAD>、<BODY>。

- <HTML></HTML>标记可理解为整个 HTML 文档的开始与结束。
- <HEAD></HEAD>标记头部，用于提供与 Web 页有关的各种信息，在<HEAD>标记中，使用<TITLE>标记来指定网页的标题。
- <BODY></BODY>标记所括住的部分是整个文档的正文。

（2）注释由开始标记<!--和结束标记-->构成，注释内容不在浏览器窗口中显示。

（3）HTML 标记不区分大小写。

【例 2-1】　HTML 示例

```
-------------------------------------------------------demo.htm-------------------------------------------------------
<html>
<head>
<title>网上教学</title>
</head>
```

```
<body>
<h3>华东交大网上教学</h3>
<h2 align= "center" >http://cai.ecjtu.jx.cn/</h2><br>
<h1>欢迎访问已有课程！</h1>
<h1>教师可申请新课程</h1>
</body>
</html>
```

用浏览器显示上述代码文件，可得到如图 2-1 所示的结果。



图 2-1　HTML 示例

【说明】

（1）网页的标题（title）在浏览器的标题栏显示。

（2）关于标记符的进一步解释。

- HTML 文档中大部分标记是成对出现的，称为成对标记，每个标记包括开始标记和结束标记，它们所括住的部分也就是标记所影响的范围；结束标记与开始标记名称相同，但结束标记总是以一个斜线符号开头（如<HTML>和</HTML>）。另外，也有部分标记只需一个符号，称为非成对标记，如
标记，其作用是实现换行。
- 大多数标记都拥有一个属性集，通过这些属性可以对作用的内容进行更多的控制。所有属性都放置在开始标记的尖括号内。本例中<h2 align= "center" >表示要使用 2 号标题且用居中方式显示文字。

2.1.2　常用 HTML 编辑工具

HTML 文件的内容均为文本，可以用任意文本编辑器进行编辑，如用 Windows 自带的记事本、写字板等，同时也可以用专门的网页制作工具进行编辑，如 Dreamweaver、FrontPage 等。使用网页制作工具的优点是速度快，可以进行图形化编辑，实现所见即所得的功能，在编辑过程中自动产生 HTML 代码。图 2-2 所示为 FrontPage 的操作界面，在该界面中单击底部的“代码”标签可切换到代码视图，查看对应的 HTML 代码。

使用网页制作工具的缺点是产生的代码格式比较混乱，可读性差，后期代码维护较困难。另外，代码中往往带有一些多余的成分，这些多余的代码增大了 HTML 文件的容量，

同时也影响了浏览器解析执行的效率。所以说，好的设计应该是文本编辑器和网页制作工具两者的结合，因此，对于 Web 开发者来说，熟悉 HTML 语言中符号的表示意义是一个必不可少的基本功。

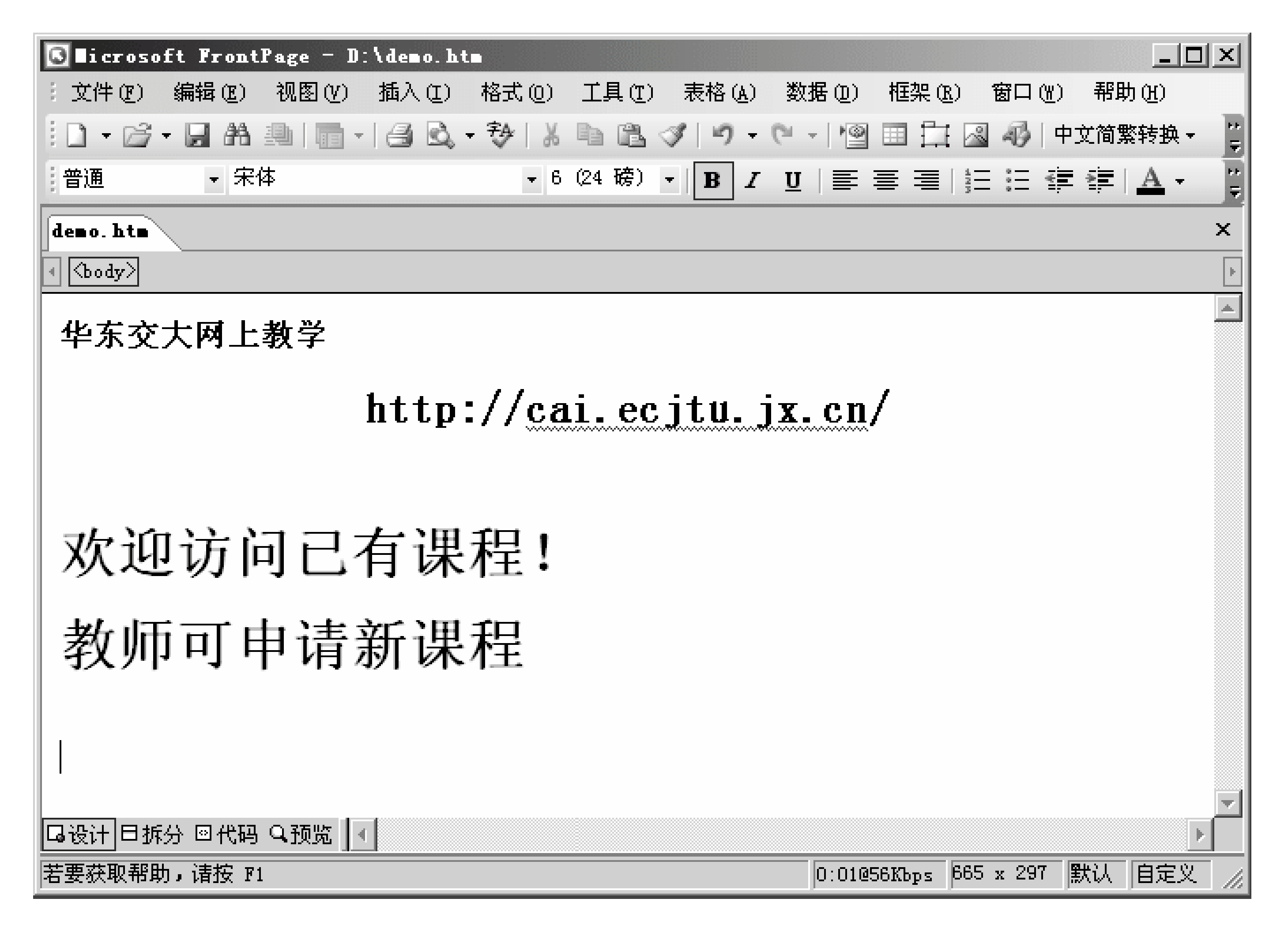


图 2-2　FrontPage 的操作界面

2.2　HTML 文本设计

2.2.1　设置 body 属性

<BODY>标记作为网页的主体部分，有很多内置属性，这些属性用于设置网页的总体风格，如表 2-1 所示。

表 2-1　<BODY>标记的主要属性

属　性	功　能
background	指定文档背景图像的 URL 地址
bgcolor	指定文档的背景颜色
text	指定文档中文本的颜色
link	指定文档中未访问过的超链接的颜色，默认为蓝色
vlink	指定文档中已被访问过的超链接的颜色，默认为紫色
alink	指定文档中正被选中的超链接的颜色，默认为红色
leftmargin	设置网页左边留出空白间距的像素个数
topmargin	设置网页上方留出空白间距的像素个数

在上述属性中，各个颜色属性的值有两种表示方法：一种是使用颜色名称来指定，如红色、绿色和蓝色分别用 red、green 和 blue 表示；一种是使用十六进制 RGB 格式表示，表示形式为 color="#RRGGBB"或 color="RRGGBB"，其中 RR 是红色、GG 是绿色、BB 是蓝色，各颜色分量的取值范围为 00～FF。例如，#00ff00 表示绿色，#FFFFFF 表示白色。

2.2.2　段落格式化

1. 分段与换行

在 HTML 文档中，不能像在 Word 中那样使用回车、空格、Tab 键来进行文档的段落调整，而是要使用标记实现分段与换行的效果。

（1）分段标记 P

分段标记用于定义段落，段落的结束由</P>来标记，</P>是可以省略的，因为下一个<P>的开始就意味着上一个<P>的结束。在默认情况下，两个段落间有一空行作为段间距。

（2）换行标记 BR

换行标记强行规定了当前行的中断，使后续内容在下一行显示。

（3）标题标记 Hn

标题标记用于设置文档中的标题和副标题，其中 n 的取值是 1～6；<H1>标记表示字体最大的标题，<H6>标记表示字体最小的标题。<H1>～<H6>标题标记会自动将字体加粗，并在文字上下空一行。

（4）水平线标记 HR

水平线标记用于在文档中添加一条水平线，用来分开文档的两个部分，该标记的属性如下。

① ALIGN：指定线的对齐方式。

② COLOR：指定线的颜色。

③ NOSHADE：若指定该项，则显示一条无阴影的实线。

④ SIZE：指定线的宽度，以像素为单位。

⑤ WIDTH：指定线的长度，单位可以是像素或百分比（占页面宽度的百分比）。

（5）预格式化标记 pre

在编辑文本中包括空格、换行、制表等排版符号，HTML 在演示内容时通常不考虑这些预格式化符号，采用预格式化标记<pre>可以保留这些符号的效果，从而使浏览器中显示的文字排列保持原来的格式。以后在处理用户从文本框提交的数据显示时，如果希望保持用户的编辑风格，可以使用该标记。

2. 设置对齐方式

在 HTML 的众多标记中，ALIGN 属性可以用来设置对齐方式，如分段标记<P>、标题标记<Hn>及水平线标记<HR>等。ALIGN 属性的取值可以是 left（左对齐）、center（居中对齐）、right（右对齐）及 justify（两边对齐）。两边对齐是指将一行中的文本在排满的情况下向左右两页边对齐，以避免在左右页边出现锯齿状。

对于不同的标记，ALIGN 属性的默认值是有所不同的。对于分段标记和<Hn>标记，ALIGN 属性的默认值为 left；对于水平线标记<HR>，ALIGN 属性的默认值为 center。

【注意】内容的对齐方式由直接包含它的标记决定，以下代码段的效果是左对齐，因为直接包含文字的标记是 h2，而 h2 默认的对齐方式是左对齐。

```
<p align=center><h2 >Web 程序设计基础教程</h2></p>
```

2.2.3　字符格式化

1．设置字体、字号和颜色

用 FONT 标记的 FACE、SIZE 和 COLOR 属性来设置文本的字体、字号和颜色。字号的取值可以是 1～7，默认值为 3，SIZE 值越大，显示的字就越大。另外，还可使用“+”或“-”号来指定相对字号，即相对默认字体大小的增量。

【例 2-2】　设置字体、字号与颜色

```
-------------------------------------------------------ex2-2.htm-------------------------------------------------------
<html>
<body>
<p><font size="4" face="宋体">Web 编程基础教程</p>
<p align=left>
<font size="+2" face="黑体" color="red">Web 编程基础教程</font>
</p>
<p align=center>
<font size="3" face="宋体" color="#00CC66">Web 编程基础教程</font>
</p>
<p><font size="6" color="#31899B">Web 编程基础教程</font></p>
<p align=right>
<font size="7" face="华文行楷" color="#CC6600">Web 编程基础教程</font>
</p>
<p align=center><h2 >Web 程序设计基础教程</h2></p>
</body>
</html>
```

运行结果如图 2-3 所示。

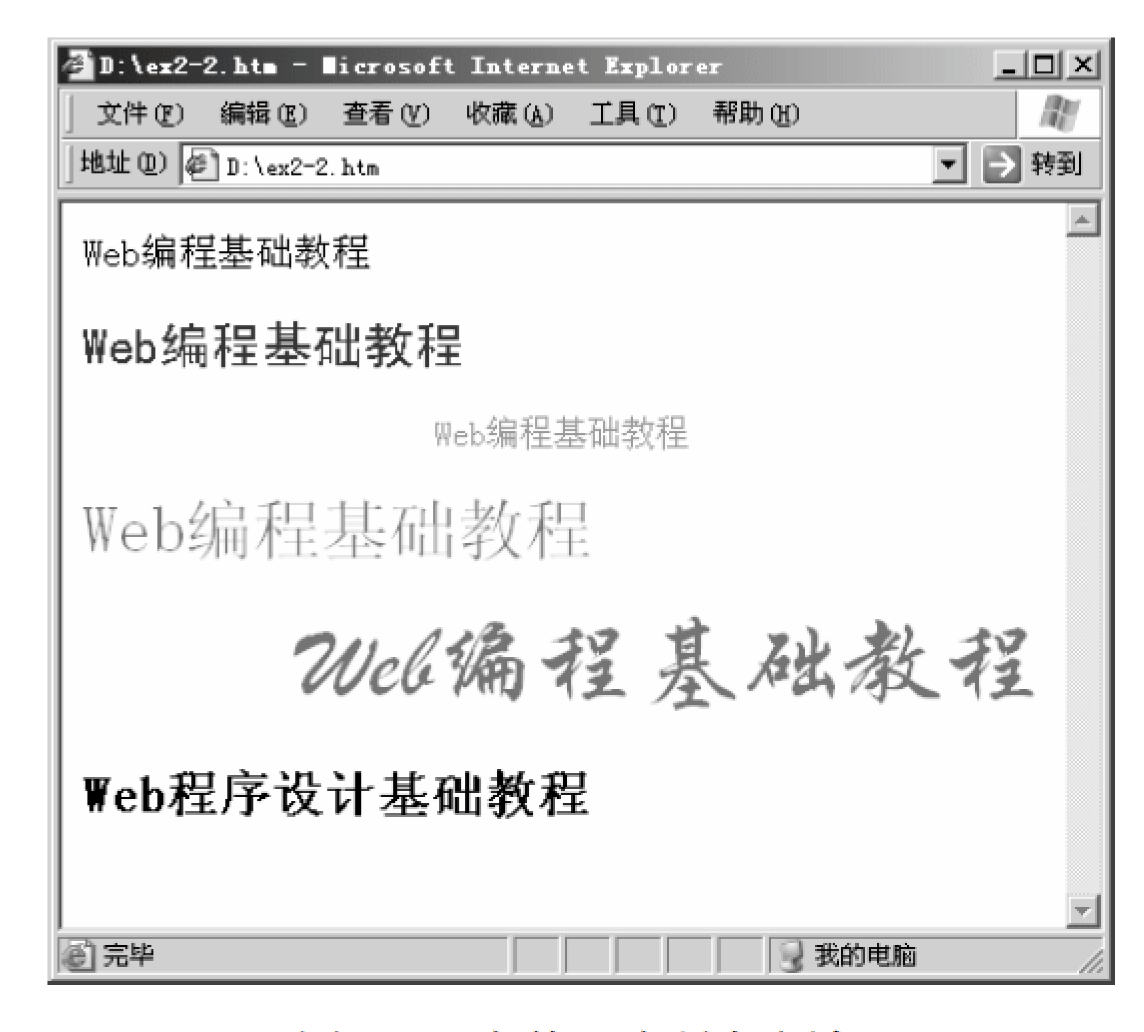


图 2-3　字体、字号与颜色

【思考】使用 size="+3"的字体相当于几号字体？

2．设置字体样式

字体样式用于为字符设置特殊格式，常用的字体样式标记如表 2-2 所示。

表 2-2　常用的字体样式标记

标　　记	字 体 样 式	标　　记	字 体 样 式
<B>…</B>	粗体	<BIG>…</BIG>	大字体
<U>…</U>	下划线	<SMALL>…</SMALL>	小字体
<I>…</I>	斜体	<STRIKE>…</STRIKE>	删除线
[…]	上标	_…	下标

3．特殊字符

诸如“<”这样的字符在 HTML 语言中具有特殊的表示意义，因此，要在网页中显示这样的字符实际是用特殊表示符实现。常用的特殊符号标记如表 2-3 所示。

表 2-3　常用的特殊符号标记

字　　符	对应字符标记	说　　明
<	&lt	小于符号
>	&gt	大于符号
&	&	AND 符号
"	&quot	双引号
		空格
•	&middot	中点
§	&sect	分节符号
©	&copy	版权符号
®	&reg	注册符号

【例 2-3】　字体样式设置

```
----------------------------------------------------------ex2-3.htm----------------------------------------------------------
<html>
<head>
<title>字体样式</title>
</head>
<body>
<p align="center"><b><big>
<font size="5" color="#0000FF" face="楷体_GB2312">作业编辑器功能说明</font></big></b>
</p>
<p><font color="#008080">编辑器允许用户设置<b>粗体</b>、<u>下划线</u>、
<i>斜体</i>、<strike>删除线</strike>，以及设置文字的<sup>上标</sup>和
<sub>下标</sub>等。</font>
</p>
<h2><font color="#FF0000">&lt;br&gt;</font>的作用是换行，
<font color="#FF0000">&lt;p&gt;</font>的作用是换段落</h2>
```

```
</body>
</html>
```

在 FrontPage 预览视图中的效果如图 2-4 所示。

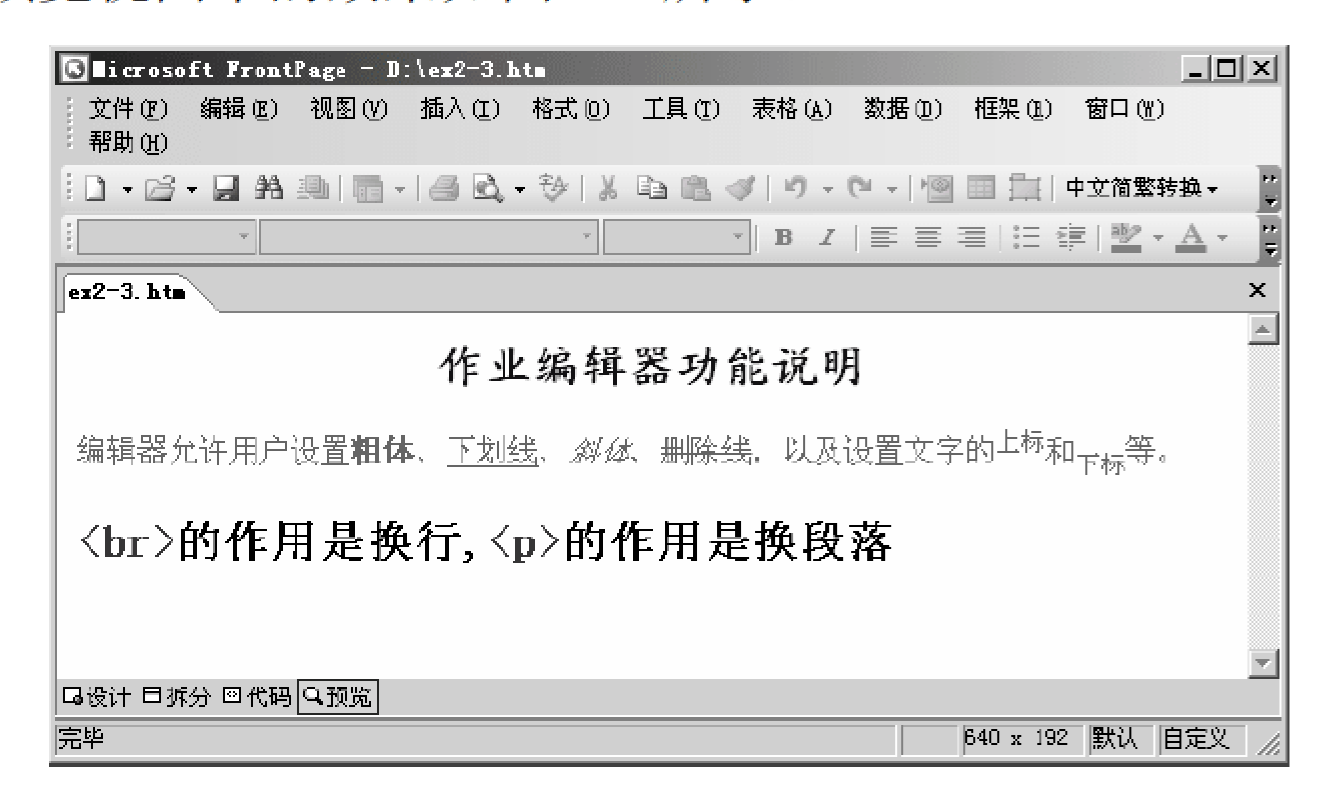


图 2-4　字体样式

【注意】当使用 FrontPage 的编辑工具时，在编辑视图中输入 HTML 标记内容，在代码视图中可看到对应的特殊符号，在预览视图中可看到设计效果。

2.2.4　使用列表格式

在网页中经常使用的列表分为有序列表和无序列表两种。由带有序号标志（如数字、字母等）的表项组成的为有序列表，否则为无序列表。

1．创建有序列表

有序列表是在各列表项前面显示数字或字母的缩排列表，可以使用有序列表标记 OL 和列表项标记 LI 来创建，语法格式如下：

```
<OL>
  <LI>列表项 1</LI>
  <LI>列表项 2</LI>
  ……
  <LI>列表项 n</LI>
</OL>
```

OL 标记有两个常用属性，即 START 和 TYPE。START 属性用于数字序列的起始值，可以取整数值；TYPE 属性用于设置数字序列样式，其取值如下。

- 1：表示阿拉伯数字 1、2、3 等，此为默认值。
- A：表示大写字母 A、B、C 等。
- a：表示小写字母 a、b、c 等。
- I：表示大写罗马数字 I、II、III、IV 等。
- i：表示小写罗马数字 i、ii、iii、iv 等。

【注意】TYPE 属性的值是区分大小写的。

当位于<OL>和</OL>标记之间时，LI 标记有两个常用属性，即 TYPE 和 VALUE。TYPE 属性指定数字样式，其取值与 OL 的 TYPE 属性相同；VALUE 属性指定一个新的数字序列起始值，以获得非连续性的数字序列。

2．创建无序列表

无序列表是一种在各列表项前面显示特殊项目符号的缩排列表，可以使用无序列表标记 UL 和列表项标记 LI 来创建，语法格式如下：

```
<UL>
  <LI>列表项 1</LI>
  <LI>列表项 2</LI>
  ……
  <LI>列表项 n</LI>
</UL>
```

UL 标记的 TYPE 属性用于指定列表项前面显示的项目符号，其取值如下。

- ❑ disc：使用实心圆作为项目符号（默认值）。
- ❑ circle：使用空心圆作为项目符号。
- ❑ square：使用方块作为项目符号。

【例 2-4】　列表项的使用

```
--------------------------------------------------------ex2-4.htm--------------------------------------------------------
<p align="left">以下演示无序列表的使用</p>
      <ul type="square">
      <li type="disc">第一章  Web 编程基础知识</li>
      <li>第二章  VBScript 脚本语言</li>
      <li type="circle">第三章  ASP  程序设计</li>
</ul>
<p>以下演示有序列表</p>
<ol>
      <li value="3">第一章  Web 编程基础知识</li>
      <li>第二章  VBScript 脚本语言</li>
      <li type="A">第三章  ASP  程序设计</li>
</ol>
```

运行结果如图 2-5 所示。

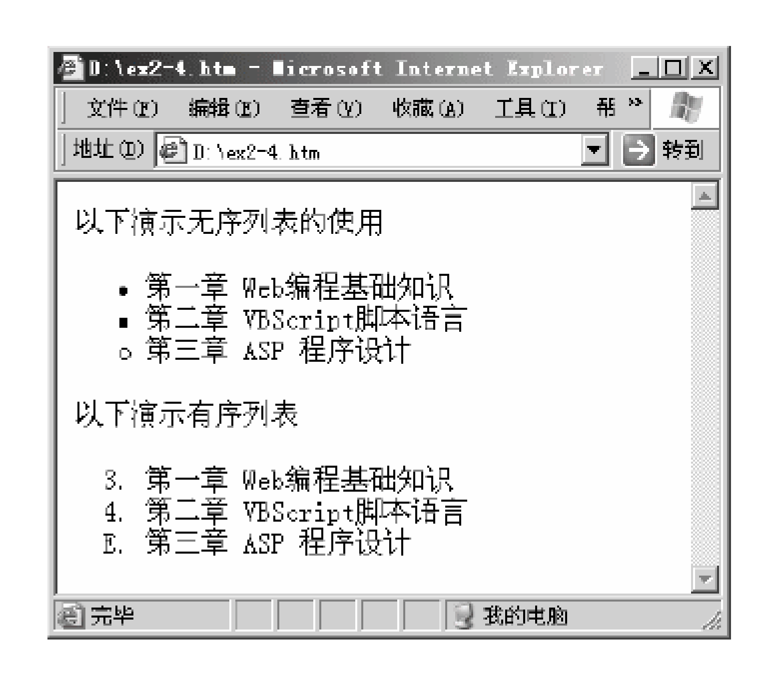


图 2-5　列表的演示

2.3 使 用 表 格

表格在网页设计中应用广泛，它不仅可作为信息的一种表示形式，还常用于页面设计中的布局与定位。

2.3.1 创建基本表格

表格一般由若干行和若干列的单元格组成，表格上面可以有一个标题，表的第一行称为表头。与表格相关的标记如下。

- <TABLE>：界定表格，最常用的属性是 BORDER，定义边界线的粗细。
- <CAPTION>：定义表格的标题。
- <TR>：定义表格的一行。
- <TH>：定义表头的各栏内容，其中字体默认为粗体。
- <TD>：定义单元格。

在 FrontPage 中，创建表格有两种常用的方法，一种是通过“表格”菜单的“插入表格”子菜单，进入“插入表格”对话框，对其中的行数、列数、高度、宽度等参数进行设置；另一种是通过“插入表格”图标，拖动鼠标选择行、列数即可。

2.3.2 表格设置

1．设置表格的属性

创建表格时，可以通过 TABLE 标记的下列属性来对表格的格式进行设置。

（1）ALIGN：指定表格的对齐方式，取值可以是 left（默认值）、center 或 right。

（2）BACKGROUND：指定用作表格背景图片的 URL 地址。

（3）BGCOLOR：指定表格的背景颜色。

（4）BORDER：指定表格边框的宽度，以像素为单位，默认值为 0。

（5）BORDERCOLOR：指定表格边框的颜色，应与 BORDER 属性一起使用。

（6）BORDERCOLORDARK：指定 3D 边框的阴影颜色，应与 BORDER 属性一起使用。

（7）BORDERCOLORLIGHT：指定 3D 边框的高亮显示颜色，应与 BORDER 属性一起使用。

（8）CELLPADDING：指定单元格内数据与单元格边框之间的间距，以像素为单位。

（9）CELLSPACING：指定单元格之间的间距，以像素为单位。

（10）WIDTH：指定表格的宽度，以像素或百分比为单位。

2．设置行的属性

表格每一行用 TR 标记定义，可以通过该标记的下列属性对指定行的格式进行设置。

（1）ALIGN：指定该行中单元格的水平对齐方式，取值为 left（默认值）、center 或 right。

（2）BACKGROUND：给出该行的背景图像文件的 URL。

（3）BGCOLOR：指定该行的背景颜色。

（4）BORDERCOLOR：指定该行的边框颜色，该属性只有当 TABLE 标记的 BORDER 属性取非零值时才起作用。

（5）BORDERCOLORDARK：指定该行的 3D 边框的阴影颜色，该属性只有当 TABLE 标记的 BORDER 属性取非零值时才起作用。

（6）BORDERCOLORLIGHT：指定该行的 3D 边框的高亮颜色，该属性只有当 TABLE 标记的 BORDER 属性取非零值时才起作用。

（7）VALIGN：指定该行中单元格内容的垂直对齐方式，该属性的取值可以是 top（顶端对齐）、middle（居中对齐）、bottom（底端对齐）或 baseline（基线对齐）。

3．设置单元格的属性

通过 TD 和 TH 标记的下列属性可以对指定单元格的格式进行设置。

（1）ALIGN：指定单元格内文本的水平对齐方式，取值为 left（默认值）、center 或 right。

（2）BACKGROUND：指定单元格背景图像的 URL。

（3）BGCOLOR：指定单元格的背景颜色。

（4）BORDERCOLOR：指定单元格的边框颜色。

（5）BORDERCOLORDARK：用于指定单元格的 3D 边框的阴影颜色。

（6）BORDERCOLORLIGHT：用于指定单元格的 3D 边框的高亮颜色。

（7）COLSPAN：指定合并单元格时一个单元格跨越的表格列数。

（8）NOWRAP：该属性可避免 Web 浏览器将单元格中的文本换行。

（9）ROWSPAN：指定合并单元格时一个单元格跨越的表格行数。

（10）VALIGN：指定单元格中文本的垂直对齐方式，取值可以是 top、middle（默认值）、bottom 或 baseline。

在 FrontPage 中，要进行表格和单元格的操作，首先要通过鼠标选取操作对象，然后通过“表格”菜单提供的功能进行操作；或者选中对象后单击鼠标右键，从弹出的快捷菜单中选择相应的功能命令完成操作。

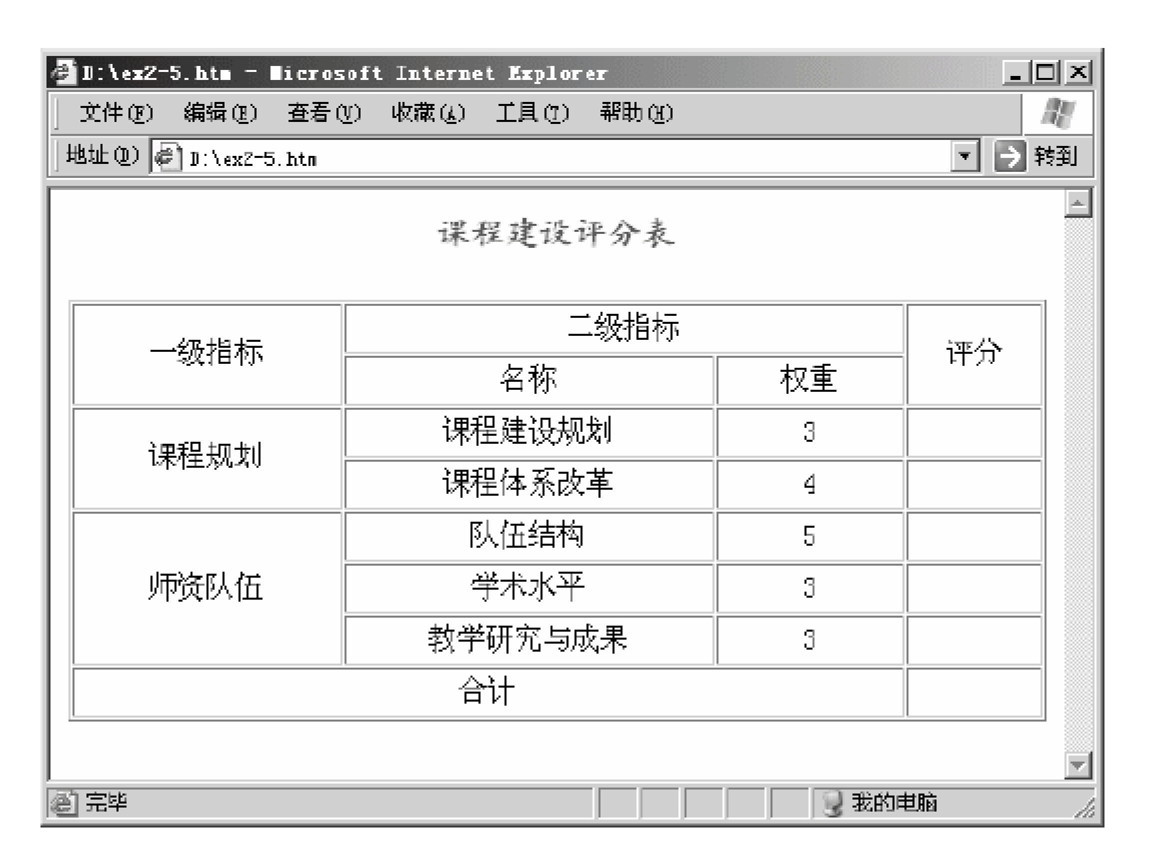
【例 2-5】　表格设计

```
-----------------------------------------------------ex2-5.htm-----------------------------------------------------
<html>
<body>
<table border="1" width="100%" cellspacing="1" >
<caption>     <!--定义表的标题  -->
<b>
<font size="4" face="楷体_GB2312" color="#008000">课程建设评分表</font>
</b><br>
```

```
</caption>
<tr>   <!--表的第 1 行 -->
<td width="213" height="28" rowspan="2" align="center">一级指标</td>
<td width="446" height="28" colspan="2" align="center">二级指标</td>
<td width="100" height="31" rowspan="2" align="center">评分</td>
</tr>
<tr>   <!--表的第 2 行 -->
<td width="296" height="28" align="center">名称</td>
<td width="144" height="28" align="center">权重</td>
</tr>
<tr> <!--表的第 3 行 -->
<td width="213" rowspan="2" align="center" height="28">课程规划</td>
<td width="296" align="center" height="28">课程建设规划</td>
<td width="144" align="center" height="28">3</td>
<td align="center">   </td>
</tr>
<tr>
<td width="296" align="center" height="28">课程体系改革</td>
<td width="144" align="center" height="28">4</td>
<td align="center">   </td>
</tr>
<tr>
<td width="213" rowspan="3" align="center" height="28">师资队伍</td>
<td width="296" align="center" height="28">队伍结构</td>
<td width="144" align="center" height="28">5</td>
<td align="center">   </td>
</tr>
<tr>
<td width="296" align="center" height="28">学术水平</td>
<td width="144" align="center" height="28">3</td>
<td align="center">   </td>
</tr>
<tr>
<td width="296" align="center" height="28">教学研究与成果</td>
<td width="144" align="center" height="28">3</td>
<td align="center">   </td>
</tr>
<tr>
<td width="665" colspan="3" align="center" height="28">
<p align="center">合计</td>
<td align="center">   </td>
</tr>
</table>
</body>
</html>
```

运行结果如图 2-6 所示。

【说明】结合代码和图的效果，分析单元格如何实现水平和垂直方向的合并。



课程建设评分表

一级指标	二级指标		评分
	名称	权重	
课程规划	课程建设规划	3	
	课程体系改革	4	
师资队伍	队伍结构	5	
	学术水平	3	
	教学研究与成果	3	
合计			

图 2-6　表格设计

2.4　在网页中加入多媒体

2.4.1　使用图像

1．在网页中插入图像

目前，在网页设计中常常大量采用图像，网页的美感大多来自精心处理的图像。可使用 IMG 标记在网页中插入一个图像。以下为 IMG 标记最常用的 4 个属性。

- ❑ SRC 属性：给出图像文件的 URL 地址，图像可以是 JPEG 文件、GIF 文件或 PNG 文件。
- ❑ ALT 属性：给出图像的简单文本说明，这段文本在浏览器不能显示图像时显示出来，或图像加载时间过长时先显示出来。
- ❑ HEIGHT 属性：设置图像的高度，所用单位可以是像素或百分数。
- ❑ WIDTH 属性：设置图像的宽度。

【注意】如果只给出了高度或宽度，则图像将按比例进行缩放。

2．设置图像格式与布局

当使用标记 IMG 在网页中插入一个图像时，可以使用 IMG 标记的如下属性对图像的格式布局进行设置。

（1）设置图像的边框

使用 IMG 标记的 BORDER 属性可以为图像添加边框效果，该属性的取值为正整数，单位为像素。例如：

```
<IMG SRC = "photo1.jpg" ALT = "校园风光" BORDER = "1">
```

（2）设置图像与文本之间的空白

使用 IMG 标记的 HSPACE 和 VSPACE 属性可以设置图像与周围文本之间的间隔，前者指定图像的左、右边距，后者指定图像的上、下边距，两者的单位均为像素。

```
<IMG SRC = "photo1.jpg" ALT = "校园风光" HSPACE = "30" VSPACE = "30">
```

（3）设置图像在页面上的对齐方式

如果插入的图像在页面上单独占一行，则可以将 IMG 标记置于<P>和</P>标记之间，并通过<P>标记的 ALIGN 属性来设置图像在页面上的对齐方式。例如：

```
<P ALIGN = "center"><IMG SRC = "photo1.jpg"" ALT = "校园风光"></P>
```

（4）设置图像与文本的对齐方式

当在页面中进行图文混排时，可以使用 IMG 标记的 ALIGN 属性来设置图像与文本在垂直方向的对齐方式，此时 ALIGN 属性的取值如下。

- top：图像与文本顶部对齐。
- middle：图像与文本中央对齐。
- bottom：图像与文本底部对齐。

通过设置 IMG 标记的 ALIGN 属性，也可以在图像的左、右绕排文本，此时 ALIGN 属性的取值如下。

- left：图像居左，文本居右。
- right：图像居右，文本居左。

【例 2-6】　图文混排

```
--------------------------------------------------------ex2-6.htm--------------------------------------------------------
<html>
<body>
<h2>    送友人（李白）</h2>
<p><img border="0" src="x.jpg" align="left" align=top>
<pre><font size="4">青山横北郭，白水绕东城。
此地一为别，孤蓬万里征。
浮云游子意，落日故人情。
挥手自兹去，萧萧班马鸣。</font></pre>
</p>
</body>
</html>
```

运行结果如图 2-7 所示。

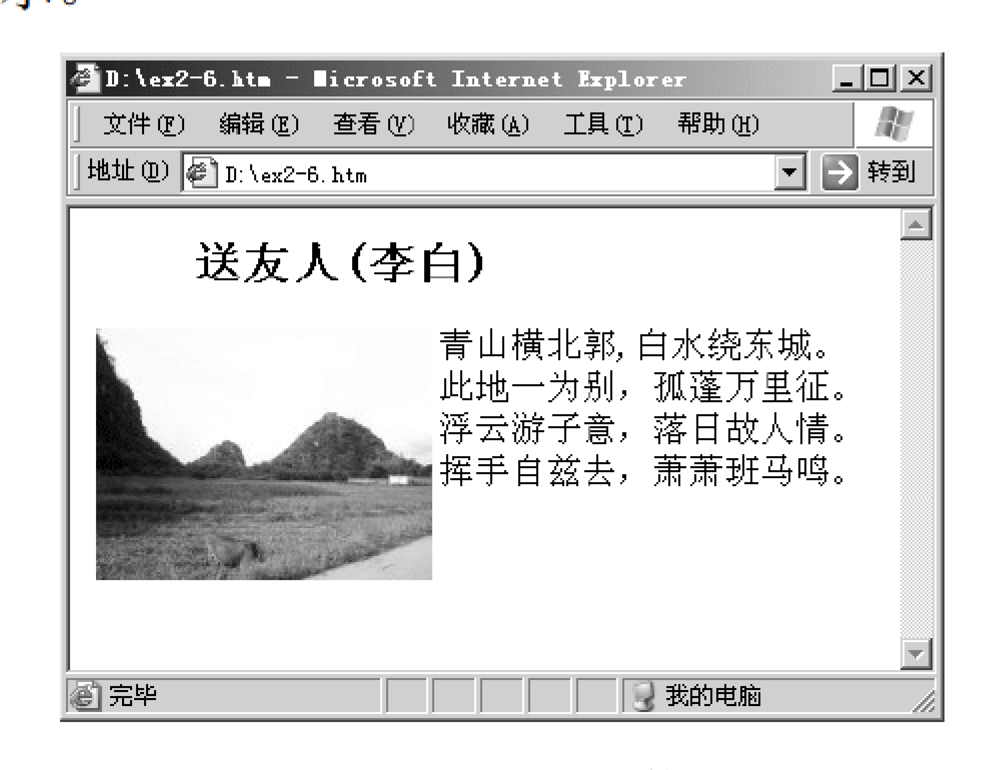


图 2-7　图文混排

【说明】此时 IMG 标记定义了两个 ALIGN 属性，一个是 left，另一个是 top。left 决定了图片在文本的左边，top 决定了文本靠顶边对齐。文本部分使用<pre>标记来保留文件中文本内容的换行风格，若去掉该标记，则文件中排版的回车符将无效，所有文字内容会按浏览器宽度进行分行显示，当一行显示不下时将自动折行。

2.4.2　使用字幕和背景音乐

1．插入字幕

MARQUEE 标记在页面中插入一个字幕，用于文本信息的滚动显示，语法格式如下：

<MARQUEE>要滚动显示的文本信息</MARQUEE>

MARQUEE 标记主要有以下属性。

（1）ALIGN：指定字幕与周围文本的对齐方式，其取值可以是 top、middle 或 bottom。

（2）BEHAVIOR：指定文本动画的类型，其取值可以是 scroll、slide 或 alternate。

（3）BGCOLOR：指定字幕的背景颜色。

（4）DIRECTION：指定文本的移动方向，其取值可以是 down、left、right 或 up。

（5）HEIGHT：指定字幕的高度，以像素或百分比为单位。

（6）HSPACE：指定字幕的外部边缘与浏览器窗口之间的左、右边距，以像素为单位。

（7）LOOP：指定字幕的滚动次数。

（8）SCROLLAMOUNT：指定字幕文本每次移动的距离，以像素为单位。

（9）SCROLLDEALY：指定与前段字幕文本延迟多少 ms 后重新开始移动文本。

（10）VSPACE：指定字幕的外边缘与浏览器窗口之间的上、下边距，以像素为单位。

【注意】MARQUEE 标记还可以定义鼠标事件，例如，在标记中添加以下属性，则鼠标移到滚动字幕时停止滚动，鼠标离开时又开始滚动。

```
<MARQUEE onMouseOver="this.stop();" onMouseOut="this.start();">
```

2．插入背景音乐

用<BGSOUND>标记在网页中添加背景音乐时，该标记只允许放在 HEAD 部分。通过<BGSOUND>标记的下列属性可以对音乐的播放进行控制。

（1）BALANCE：指定如何将声音分成左声道和右声道，取值为-10000～+10000，默认值为 0。

（2）LOOP：指定声音播放的次数。如果设置为 0，则播放一次；如果设置为大于 0，则播放指定的次数；如果设置为-1，则声音反复播放，直到页面卸载为止。

（3）SRC：指定要播放的声音文件的 URL。常用的声音文件类型是波形文件（.wav）、MIDI 文件（.mid）、AIFF 文件（.aif）、AU 文件（.au）及 MP3 文件（*.mp3）等。

（4）VOLUME：指定音量高低，取值为-10000～0，默认值为 0。

2.5　使 用 框 架

2.5.1　框架网页的基本结构

框架网页将浏览器上的视窗分成不同的区域，在每个区域中都可以独立显示一个网页。框架网页通过一个或多个 FRAMESET 和 FRAME 标记来定义。FRAMESET 表示框架集，FRAME 表示一个框架，也常称作帧。在框架网页中，将 FRAMESET 标记置于 HEAD 之后，以取代 BODY 的位置，另外，还可以使用 noframes 标记给出框架不能被显示时的替换内容。

框架网页的基本结构表示如下：

```
<HTML>
<HEAD>
<TITLE>框架网页的基本结构</TITLE>
</HEAD>
<FRAMESET>
    <FRAME>
    <FRAME>
    ……
   <noframes>
    <body>
    <p>此网页使用了框架，但您的浏览器不支持框架。</p>
    </body>
    </noframes>
</FRAMESET>
</HTML>
```

【说明】当以上格式中的某个<FRAME>标记为<FRAMESET>时，就形成了嵌套的框架。也就是说，在框架集中又包含了子框架集。

在 FrontPage 中创建框架的步骤是：（1）选择“文件”→“新建”→“其他网页模板”命令，将得到如图 2-8 所示的“网页模板”对话框。（2）从该对话框的“框架网页”选项卡中选择某个模板，在右下方的“预览”位置即可看到预览效果，然后单击“确定”按钮。

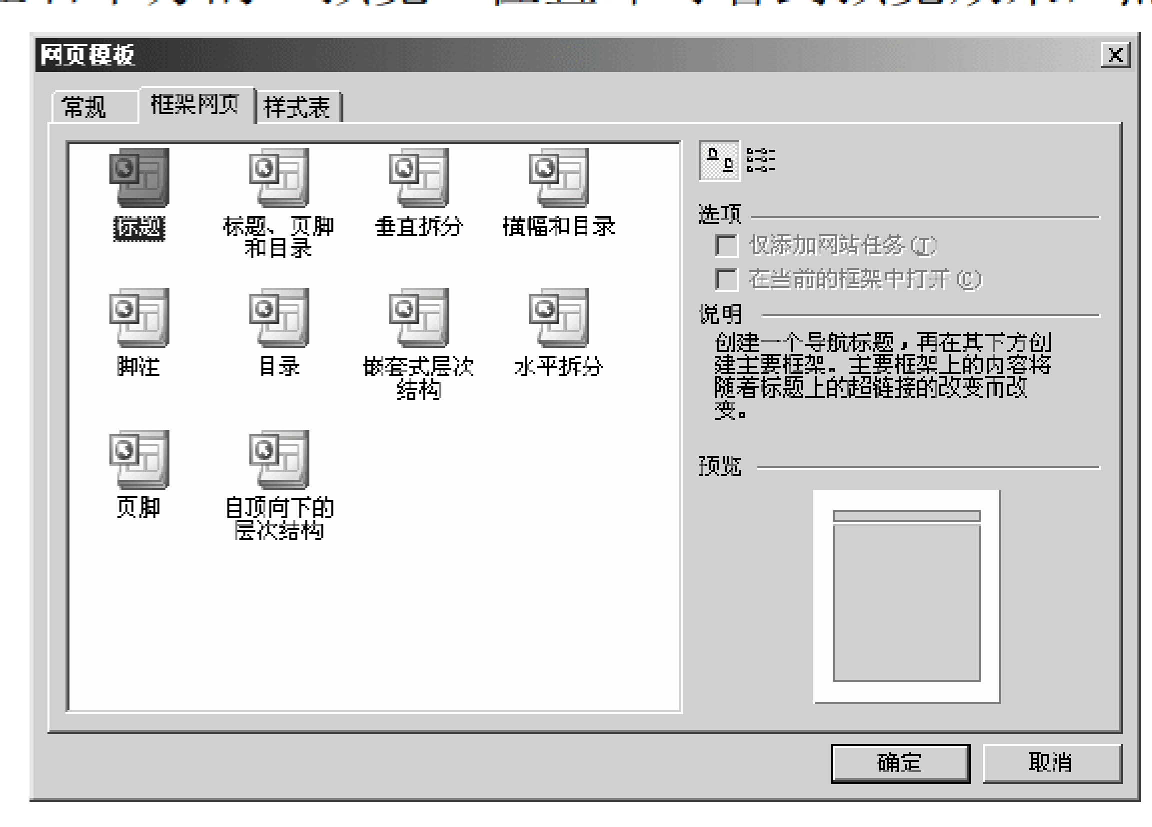


图 2-8　“网页模板”对话框

2.5.2　框架的设置

1．设置框架集的属性

框架集包含了如何组织各个框架的信息，框架是按照行或列来组织的，可以使用 FRAMESET 标记的下列属性对框架的结构进行设置。

（1）COLS：在创建纵向分隔框架时，通过该属性指定各个框架的列宽，取值有 3 种形式，即像素、百分比（%）和相对尺寸（*）。

（2）FRAMEBORDER：指定框架周围是否显示三维边框，如果取值为 1（默认值）或 yes，则表示显示三维边框；如果取值为 0 或 no，则表示显示平面边框。

（3）FRAMESPACING：指定框架之间的间隔，以像素为单位。如果不设置该属性，则框架之间没有间隔。

（4）ROWS：在创建横向分隔框架时，通过该属性指定各个框架的行高，取值有 3 种形式，即像素、百分比（%）和相对尺寸（*），设置方法与 COLS 属性类似。但需要注意的是，在同一个框架集内，ROWS 属性不能与 COLS 属性同时使用，若要创建同时包含纵向分隔框架和横向分隔框架，则应使用嵌套框架，也就是在框架集中又包含有框架集。

2．设置框架的属性

每个框架的属性都是由<FRAME>标记设置的，包括框架的名称、框架是否可以滚动以及在框架中显示什么文件等。<FRAME>标记具有下列属性。

（1）FRAMEBORDER：指定框架周围是否显示三维边框，取值为 1 或 yes 表示显示三维边框；取值为 0 或 no 表示显示平面边框。默认值为 1。

（2）MARGINHEIGHT：指定框架的高度，以像素为单位。

（3）MARGINWIDTH：指定框架的宽度，以像素为单位。

（4）NAME：指定框架的名称。

（5）NORESIZE：若指定了该属性，则将无法调整框架的大小。

（6）SCROLLING：指定框架是否可以滚动。若该属性设置为 yes，则框架可以滚动；若该属性设置为 no，则框架不能滚动；若该属性设置为 auto，则在需要时自动添加滚动条。

（7）SRC：指定在框架中显示的网页文件的 URL 地址。

【例 2-7】　Java 语言教学课件的框架设计

```
---------------------------------------------------------default.htm---------------------------------------------------------
<html>
<head>
<title>Java 网络课件</title>
</head>
<frameset frameborder=0   rows="103,*"   border=0 name="topframe">
  <frame src="mytop.htm"   SCROLLING="no" name="top" >
  <frameset frameborder=0 cols="185,8,*" border=0 name=mainframe >
     <frame src="chapmenu.htm" NAME="nav" SCROLLING="auto" noresize>
     <frame SRC="left.htm" NAME="control"   SCROLLING="no" noresize>
```

```
      <frame SRC="welcome.html" NAME="main" SCROLLING="auto" noresize>
   </frameset>
</frameset>
</html>
```

访问该网站的效果图如图 2-9 所示。

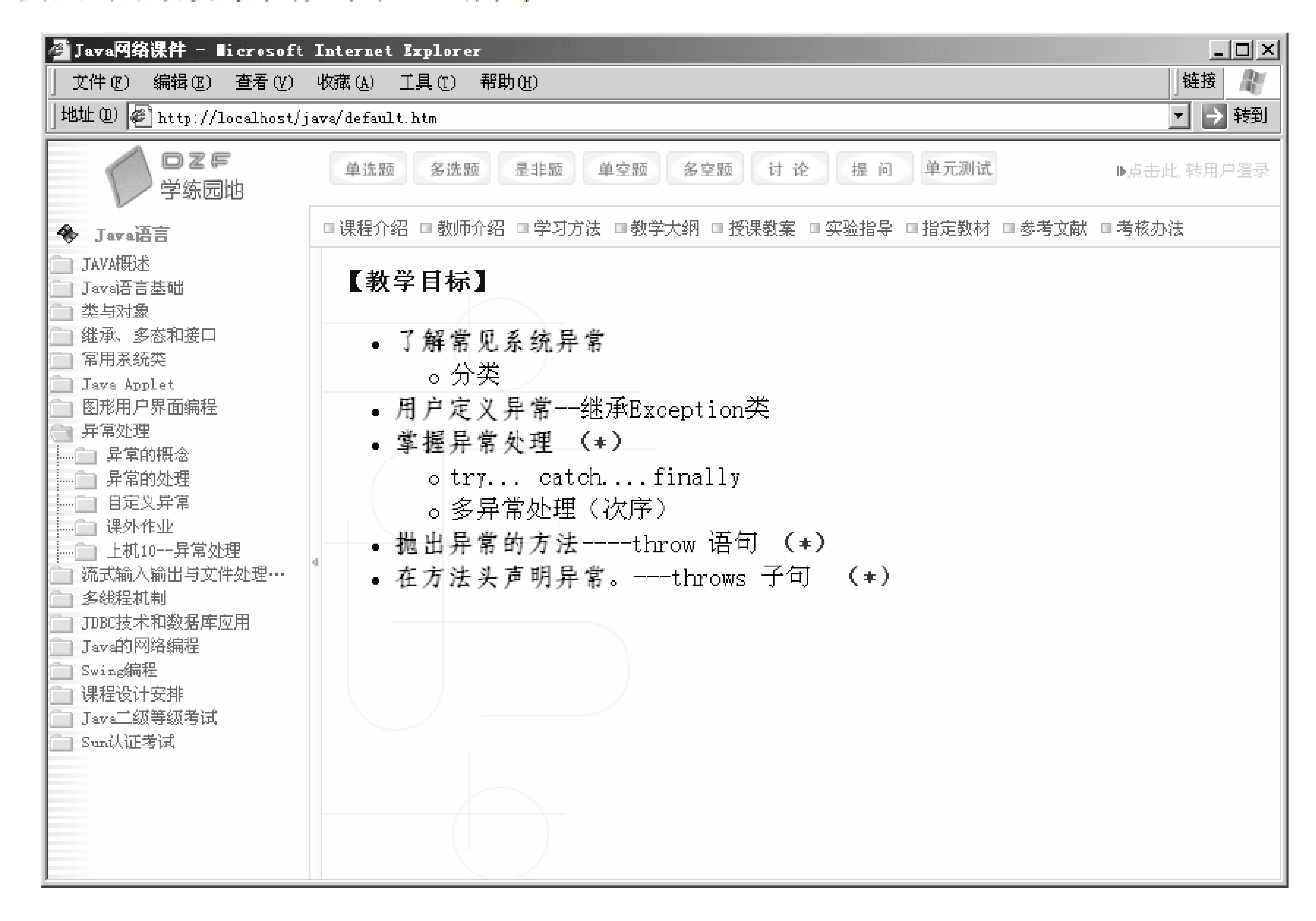


图 2-9　Java 教学课件学生学习浏览界面

【说明】该界面采用了嵌套的框架集，最外层的框架集分为上、下两部分，顶部一帧用来显示网站徽标和功能图标；下面为一个框架集，由 3 帧构成，左侧一帧用来显示章节列表，也就是章节导航选择菜单，中间有一个很窄的帧用来控制是否让整个窗体均显示教学内容，右侧一帧用来显示教学内容。该网站在 Internet 上的网址为 http://cai.ecjtu.jx.cn/java/default.htm。

2.6　使用超链接

2.6.1　理解超链接和路径

超链接是由源端点到目标端点的一种跳转。源端点可以是网页中的一段文本或一幅图像等，而目标端点可以是任意类型的网络资源，如一个网页、一幅图像等。

按照目标端点的不同，可以将超链接分为以下几种形式。

（1）文件链接：这种链接的目标端点是一个文件，它可以位于当前网页所在的服务器，也可以位于其他服务器。

（2）锚点链接：这种链接的目标端点是网页中的一个位置，通过这种链接可以从当前网页跳转到本页面或其他页面中的指定位置。

（3）E-mail 链接：通过这种链接可以启动电子邮件客户端程序（如 Outlook 或 Foxmail 等），并允许访问者向指定的地址发送邮件。

路径用来指定超链接中目标端点的位置，通常有以下几种类型。

（1）绝对路径：给出了目标文件的完整 URL 地址，包括传输协议在内。如果要链接的文件位于外部服务器上，则必须使用绝对路径。

（2）相对路径：是指以当前文档所在位置为起点到目标文档所经过的路径。若要将当前文档与处在同一文件夹中的另一个文档链接，或者将同一站点中不同文件夹下的文档相互链接，都可以使用相对路径。

2.6.2　创建文件链接

文件链接是超链接中最常用的一种链接形式，其基本语法格式如下：

<A HREF = "字符串" TARGET = "字符串" TITLE = "字符串">显示文本</A>

其中各属性描述如下。

（1）HREF：该属性是必选项，用于指定目标端点的 URL 地址。

（2）TARGET：该属性是可选项，用于指定一个窗口或框架的名称，目标文档将在指定窗口或框架中打开。如果省略该属性，则在超链接所处的窗体或框架中打开目标文档。TARGET 属性的取值及作用如表 2-4 所示。

表 2-4　TARGET 属性的取值及作用

取　值	作　用
name	在指定名称的窗口或框架中加载目标文件
_blank	在未命名的新浏览器窗口中加载目标文件
_parent	在父框架页或窗口中加载目标文件
_self	在同一框架或窗口中加载目标文件
_top	将目标文件加载显示占据整个浏览器窗口

需要特别注意的是，在网页内通过<base>标记可规定默认在哪个目标框架显示超链接的文档。例如，在某页面中设置<base target="main">，则该页面中所有没用 TARGET 属性指定目标框架的超链接将在名字为 main 的框架中显示目标文档。

（3）TITLE：该属性是可选项，用于指定鼠标移到超链接上时所显示的标题文字。

2.6.3　创建锚点链接

在创建锚点链接时，应在页面的某处设置一个位置标记（即所谓锚点），并为该位置指定一个名称，以便在同一页面或其他页面中引用。通过创建锚点链接，可以使超链接指向当前页面或其他页面中的指定位置。

若要创建锚点链接，则首先应在页面中通过 A 标记定义一个锚点（也称书签），通过 NAME 属性为其命名，但不要在<A>和</A>标记之间放置任何文字。例如，在 test.htm 中

安排：

```
<A NAME = "top"></A>
```

创建锚点后，可以使用 A 标记来创建指向该锚点的超链接。例如，要在同一个页面中跳转到名为 top 的锚点处，可以使用以下 HTML 代码：

```
<A HREF = "#top">返回顶部</A>
```

要在其他页面中跳转到该锚点，则可以使用以下 HTML 代码：

```
<A HREF = "test.htm#top">跳转到 test.htm 页的顶部</A>
```

2.6.4　创建邮件链接

使用 A 标记创建邮件链接，其 HREF 属性应由 3 部分组成：第一部分是电子邮件协议名称 mailto，第二部分是电子邮件地址，第三部分是可选的邮件主题，其形式为“subject=主题”。第一部分与第二部分之间用冒号（:）分隔，第二部分与第三部分之间用问号（?）分隔。例如：

```
<A HREF = "mailto:zfding@ecjtu.jx.cn?subject=小问题">给我写信</A>
```

当访问者在浏览器窗口中单击邮件链接时，将会自动启动电子邮件客户端程序，并将指定的主题填入“主题”栏中。

2.7　使用表单

表单（Form）是动态网页编程中使用最多的 HTML 元素，它是网页实现交互功能的主要接口，用户通过它向服务器提交数据。表单中可以包含允许用户进行交互的各种控件，如文本框、列表框、复选框和单选按钮等。

2.7.1　表单处理概述

1．表单的基本语法

表单用 FORM 标记定义，其语法格式如下：

```
<FORM NAME="字符串" METHOD="get|post" ACTION="字符串" TARGET="字符串">
……
</FORM>
```

FORM 标记具有以下属性。

（1）NAME：指定表单的名称，以标识表单。

（2）METHOD：指定将表单数据传输到服务器的方法。

（3）ACTION：指定将要接收表单数据的服务器端程序或动态网页的网址。

（4）onSubmit：指定提交表单时调用的事件处理程序。

（5）onReset：指定重置表单时调用的事件处理程序。

（6）TARGET：指定表单提交后打开新文档的目标窗口，其取值可参考表 2-4。

2．提交和处理表单

当用户填完表单数据后，单击“提交”按钮即可将表单数据提交给 Web 服务器上的表单处理程序。FORM 标记的 METHOD 属性确定提交方法，有 get 和 post 两种情形。FORM 标记的 ACTION 属性确定表单处理程序的 URL 地址，在该程序中可以读取来自表单的数据。

如果要在客户端处理表单，则要借助事件处理程序，通过 onSubmit 属性定义表单提交时要执行的客户端脚本函数或代码。一般情况下，常用该方法在客户端进行数据校验，校验通过后才将数据提交给服务器，从而防止非法数据提交给 Web 服务器。

2.7.2　INPUT 标记型表单控件的使用

将 INPUT 标记的 TYPE 属性设置为不同的值，可以创建不同类型的输入型表单控件，包括单行文本框、密码框、复选框、单选按钮、文件域及按钮等。每个控件均有一个 NAME 属性或 ID 属性，用于指定该控件的名称或标识，以便在脚本代码中引用该控件。

1．单行文本框

单行文本框用于获取一行信息，其基本语法格式如下：

<INPUT TYPE = "text" NAME = "字符串" VALUE = "字符串" SIZE = "整数" MAXLENGTH = "整数">

其中，SIZE 属性指定文本框的宽度；MAXLENGTH 属性指定允许在文本框中输入的最大字符数；VALUE 属性指定文本框的默认值，如果不设置默认值，可省略该属性。

2．密码域

在密码域中输入数据时，浏览器以星号显示密码。其基本语法格式如下：

<INPUT TYPE="password" NAME="字符串" VALUE="字符串" SIZE="整数" MAXLENGTH= "整数">

其中，VALUE 属性用于指定密码域的初始值；SIZE 属性用于指定密码域的宽度；MAXLENGTH 属性用于指定允许在密码域中输入的最大字符数。

3．按钮

用<INPUT>标记可以在表单中添加 3 种类型的按钮，即提交按钮、重置按钮和自定义按钮。创建按钮的基本语法格式如下：

<INPUT TYPE="submit|reset|button" NAME="字符串" VALUE="字符串" >

对该标记的属性说明如下。

（1）TYPE：指定按钮的类型，取值如下。

- submit：创建一个提交按钮。

- reset：创建一个重置按钮。
- button：创建一个自定义按钮。

（2）VALUE：指定显示在按钮上的标题文本。

4．图形化按钮

在表单中可以用图片作为提交按钮，从而起到美化页面的设计效果。图形化按钮的基本语法格式如下：

<INPUT TYPE="image" SRC="URL" NAME="字符串" VALUE="字符串">

其中，TYPE="image"表示以一个图像作为提交按钮；SRC 属性给出图像的 URL 地址；VALUE 属性提供图像的替换文本。

5．复选框

复选框用于从可选项中选择 0 到多个选项，其基本语法格式如下：

<INPUT TYPE="checkbox" NAME="字符串" VALUE="字符串" [CHECKED]>选项文本

其中，VALUE 属性指定提交时的值；CHECKED 属性是可选的，若使用该属性，则表示该复选框默认为选中状态。

在提交表单时，如果复选框被选中，则其内部名称和值都会包含在表单结果中。对于未选中的复选框，只有名称会被纳入表单结果中，但值为空白。

【注意】对于同一标识名的复选框，由于有多个值可能被选中，因此在提交表单时，该名称表单域的值是一个集合。

6．单选按钮

单选按钮用于从一组选项中选择其中之一，其基本语法格式如下：

<INPUT TYPE="radio" NAME="字符串" VALUE="字符串" [CHECKED]>选项文本

其中，若干个名称相同的单选按钮构成一个控件组，在该组中只能选中一个选项。VALUE 属性指定提交时的值；CHECKED 属性是可选的，若某个单选按钮拥有该属性，则打开表单时该单选按钮处于选中状态。

在提交表单时，在同一名称的单选按钮的读取结果中只含有被选中单选按钮的值，如果没有任何单选按钮被选中，则值为空白。

7．文件域

文件域用于上传文件到服务器，它由一个文本框和一个“浏览”按钮组成。用户既可以在文本框中输入文件的路径和文件名，也可以通过单击“浏览”按钮从磁盘上查找和选择所需文件。文件域的基本语法格式如下：

<INPUT TYPE="file" NAME="字符串" SIZE = "整数" VALUE = "字符串">

其中，VALUE 属性给出初始文件名，SIZE 属性指定文件名输入框的宽度。

8．隐藏域

要在表单中包含不希望站点访问者看见的信息，则可以在表单中添加隐藏域。隐藏域常用于在相邻页面间传递特殊的信息，其基本语法格式如下：

<INPUT TYPE="hidden" NAME="字符串" VALUE="字符串">

其中，VALUE 属性给出隐藏域的默认值。当提交表单时，该隐藏域的名称和值就会与可见表单域的名称和值一起包含在表单结果中。

【注意】如果用户查看浏览器的页面代码，则依然可以看到隐藏域和它的值。

2.7.3　其他表单控件

1．滚动文本框

滚动文本框用于输入多行文本，也称多行文本框或称文本域，其基本语法格式如下：

<TEXTAREA NAME="字符串" ROWS="整数"COLS="整数" [READONLY]>

……

</TEXTAREA>

其中，ROWS 属性指定该控件的高度（以行为单位）；COLS 属性指定该控件的宽度（以字符为单位）；READONLY 属性指定滚动文本框的内容不被用户修改。

<TEXTAREA>和</TEXTAREA>标记之间的文本将作为该控件的初始值。

2．下拉列表框

下拉列表框用于从多个列表项中选择一个或多个，其基本语法格式如下：

<SELECT NAME = "字符串" SIZE = "整数" [MULTIPLE]>

<OPTION [SELECTED] VALUE = "字符串">选项 1</OPTION>

<OPTION [SELECTED] VALUE = "字符串">选项 2</OPTION>

……

</SELECT>

其中，SIZE 属性指定在列表中一次可以看到的选项数目；MULTIPLE 指定是否允许作多项选择，如果无该选项，则只能选择一项；<OPTION>标记将每个选项列出来，其中，SELECTED 属性指定该选项的初始状态为选中。

3．表单控件分组

为了美化表单控件的页面布局，可以使用<FIELDSET>标记对表单控件进行分组。注意，<FIELDSET>标记必须以<LEGEND>开头，以提供控件组的标题，在<LEGEND>之后可以跟其他表单控件，也可以嵌套<FIELDSET>。其基本语法格式如下：

<FIELDSET>

<LEGEND>控件组标题</LEGEND>

组内表单控件

</FIELDSET>

【例 2-8】　表单控件分组的应用

```
--------------------------------------------------------ex2-8.htm--------------------------------------------------------
<html>
<body>
<center>
```

```
<FIELDSET   style="left: .9em; top: 26.1em; width: 513px; height: 264px;">
<LEGEND style="color:green;">课程访问设置...</LEGEND>
<form action="systemsetupwrite.asp" method="POST">
<table border="0" width="100%" height="163">
<tr>
<td align="right">
<img border="0" src="images/edit1.gif" width="16" height="16"></td>
<td align="center" width="156">学生注册批准方式？</td>
<td align=CENTER width="60%">
     <table border="0" width="76%" >
     <tr><td ALIGN=center height="40" >自动批准
     <input type="radio" name="allow" value="True"    checked ></td>
     <td ALIGN=center>教师批准
     <input type="radio" name="allow"   value="False" >
     </td>
     </table>
</td>
</tr>
……为节省篇幅，此处省略了部分代码，读者可参照图补充……
</table>
<br>
<input type="submit" name="b1" value="提   交">
</form>
</FIELDSET>
</center>
</body>
</html>
```

在网络教学平台中该程序的运行结果如图 2-10 的右下帧所示。

【说明】本例是网络教学系统中课程访问设置表单，实际应用中表单各控件的设置值来自数据库，这里将其显示内容静态化为 HTML 程序。FIELDSET 标记中的 Style 属性的定位样式设置表单分组控件在页面中的显示位置。

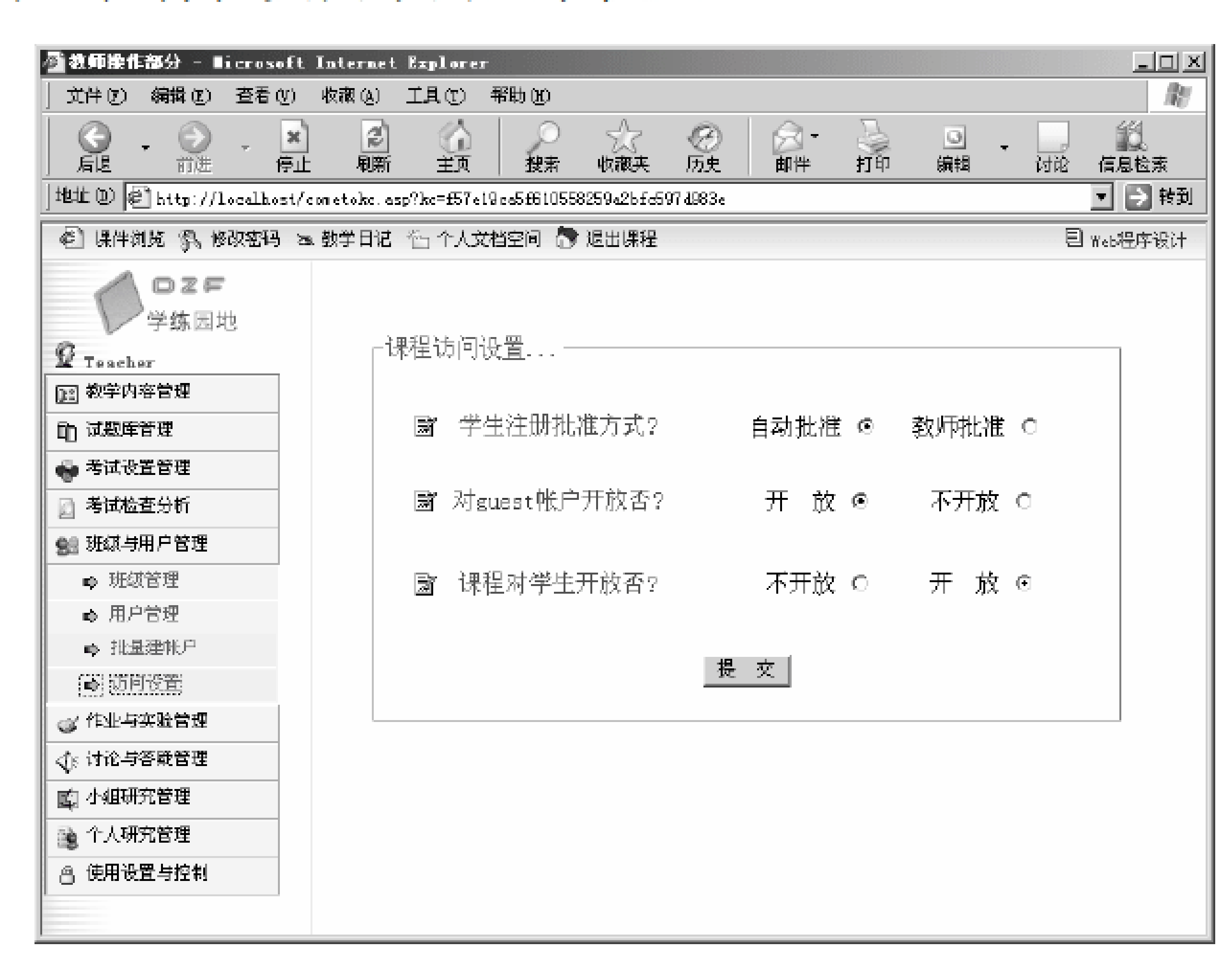


图 2-10　表单控件分组的应用

本 章 小 结

HTML 是网页设计的基本语言，熟悉常用 HTML 标记的使用是 Web 编程的一项基本功。本章介绍了 HTML 语言常用标记的功能和应用特点。在实际应用中虽然可借助 FrontPage 和 Dreamweaver 等工具制作网页，并自动生成各类标记，但由于 ASP 程序与网页内容是混合在一起的，因此在 Web 编程中用户必须对各类标记有足够的了解，这样才能正确、灵活地进行编程处理，并根据需要合理规划和部署网页内容。

习　　题

1．选择题

（1）若要限制在单行文本框中所能输入的最多字符数，应使用<INPUT>标记的（　　）属性。

A．SIZE　　B．VALUE　　C．TABINDEX　　D．MAXLENGTH

（2）下列表单字段中适合作为单一的选择题使用的是（　　）。

A．单行文本框　　B．复选框　　C．单选按钮　　D．下拉列表框

（3）下列表单字段中适合用来输入网上招聘的自我介绍内容的是（　　）。

A．复选框　　B．多行文本框　　C．单行文本框　　D．下拉列表框

（4）有关单选按钮的语句，下面说法错误的是（　　）。

A．单选按钮适合用来询问只有一个答案的问题

B．同一组单选按钮的每个单选按钮的名称必须相同

C．<INPUT>标记的 TYPE 属性需设置为 CHECKBOX

D．同一组单选按钮的每个单选按钮通过 VALUE 属性区分

（5）若要设置下拉列表的各个选项，可以使用的标记是（　　）。

A．<OPTION>　　B．<LI>　　C．<SELECT>　　D．<OL>

（6）在 HTML 中，标记<pre>的作用是（　　）。

A．标题标记　　B．预排版标记　　C．转行标记　　D．文字效果标记

（7）若要将表单数据以字符串的方式附加在网址的后面返回服务器端，必须将<FORM>标记的 METHOD 属性设置为（　　）。

A．POST　　B．GOT　　C．GET　　D．QUERY

（8）在不指定特殊属性的情况下，可以手动输入文本的 HTML 标签的是（　　）。

A．<TEXTAREA></TEXTAREA>　　B．<INPUT type="text"/>

C．<INPUT type="hidden"/>　　D．<DIV></DIV>

（9）对网页标题的描述，下述说法正确的是（　　）。

A．<title>…</title>之间的内容为网页标题

B. <head>…</head>之间的内容为网页标题

C. 网页标题一定要有，否则网页显示不出来

D. 网页标题内容会在网页左下角以滚动形式显示出来

2. 制作题

（1）利用表格标记制作本学期的课程表。

（2）制作一个本班同学录的页面，表格数据项包括姓名、地址、联系电话、电子邮件等，表头用灰色背景，表格内的文字用红色，表格的边框用绿色。

（3）用表格进行定位，制作一个如图 2-11 所示的组卷选题设置页面。



图 2-11　组卷选题设置页面

（4）设计一个个人主页，该主页中包括个人基本信息，如姓名、出生年月、性别、籍贯以及成长简历、照片等，用表格进行布局设计。另有一些信息通过超链接的形式增加，如个人兴趣、主要业绩等可安排在专门的页面中用列表项列出。

（5）编写一个框架网页，在顶部创建一个用于显示网站标题的横幅框架，在左边创建一个用于显示网页链接的目录框架，并在右下方创建一个用于显示内容的主要框架。要求在目录框架中单击不同链接时，在主要框架中显示相应的网页。

第 3 章　VBScript 介绍

3.1　VBScript 概述

VBScript 是一个以 Visual Basic 语言为基础的轻量级程序语言，它是 Visual Basic 语言的一个子集。最初，VBScript 用于编写客户端事件驱动代码以扩大 HTML 的功能，如在客户端进行输入数据的有效性验证，防止浏览器将无效数据发送给服务器。随着 ASP 技术的出现，VBScript 将它的功能扩展到了服务器上，目前，VBScript 的主流应用是编写服务器端的脚本代码。因此，本章介绍的内容主要是 VBScript 的语法和函数，将略去用 VBScript 进行客户端事件编程的内容，客户端的事件编程通常使用 JavaScript，这将在后续章节中进行介绍。

在服务器端的 ASP 程序中可使用<%和%>作为识别标记符来插入 VBScript 代码，这些代码将在 Web 服务器端解释执行。

在网页中安排在客户端执行的 VBScript 代码，应使用<script>标记。有两种方式：一种方式是将该标记的 type 属性赋值为"text/vbscript"；另一种方式是使用 script 标记的 language="vbscript"，但 language 这个属性在 W3C 的 HTML 标准中已不再推荐使用。

【例 3-1】　浏览器端的 VBScript 代码

```
-------------------------------------------------------ex3-1.htm-------------------------------------------------------
<html>
<head><title>简单的 VBScript 示例</title></head>
<body>
<p>你会看到一个消息框，在此行文字显示之后弹出。</p>
<script type="text/vbscript">
    MsgBox("Hello,World!")
</script>
</body>
</html>
```

在浏览器页面中将弹出一个对话框，如图 3-1 所示。

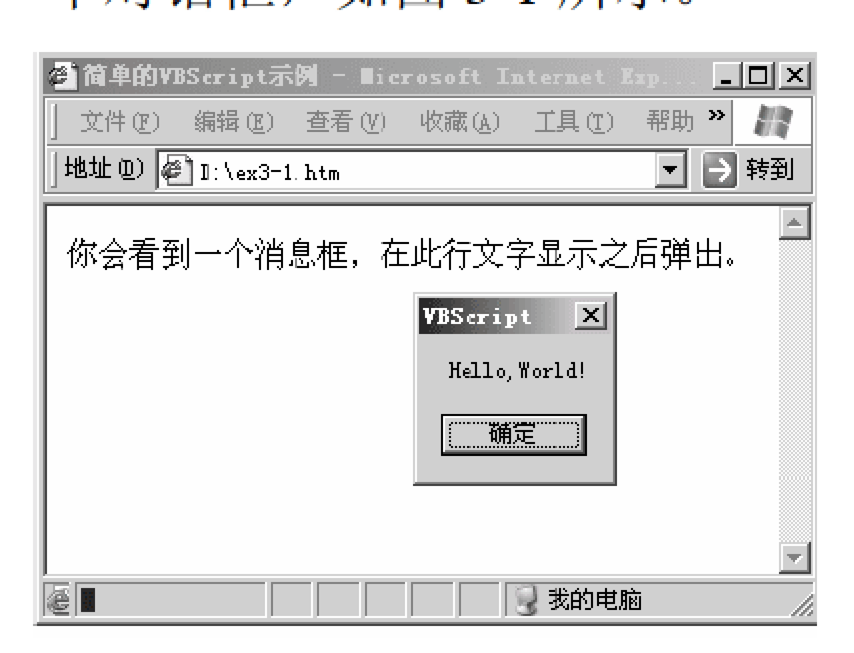


图 3-1　VBScript 示例

【说明】此时，MsgBox 为 VBScript 的一个内部函数，用于在客户端弹出消息框显示信息，详见本章第 3.5 节的相关内容。

在客户端代码中，VBScript 可以放在网页的 Body 或 Head 中。一般将函数定义的代码集中放置在 Head 中。

3.2　VBScript 的数据表示

3.2.1　VBScript 的数据类型

VBScript 中只有一个基本数据类型，即 Variant，也称为变体类型，但变量可根据所赋值的不同而代表不同类型。对于 Variant 来说，当其用于数字上下文中时作为数字处理，而当其用于字符串上下文中时作为字符串处理。除数字和字符串外，VBScript 中还包括布尔类型、时间类型等，它们被称为数据子类型。表 3-1 中列举了 VBScript 中的数据子类型。

表 3-1　VBScript 中的数据子类型

子　类　型	描　　述
Boolean	布尔值，其值是 True 或 False
Byte	0～255 之间的整数
Integer	整数
Long	长整数
Single	单精度浮点数
Double	双精度浮点数
Date	表示日期的数值，用#号括起来，如#2001-4-10#、#2003-9-12 8:20:10 AM#
String	变长字符串，最大长度可为 20 亿个字符
Object	对象类型，用来引用程序所能识别的任何对象

3.2.2　VBScript 的常量、变量与数组变量

1．常量

常量的值是固定的，它可以用来代表字符串、数字和日期等常数，一经声明，其值将不能再更改。声明常量时使用了 CONST 语句，例如：

```
CONST pi=3.14159
```

这里，常量 pi 被分配了值 3.14159。在脚本中将不能给 pi 分配新值。

VBScript 定义了一些固有常量，如表 3-2 所示。

表 3-2　VBScript 的固有常量

常 量 名 称	常 量 含 义
True	布尔类型的真值
False	布尔类型的假值
Empty	声明一个变量，如果为该变量赋值，则该变量的值为 Empty。对于数值变量，其值为 0；对于字符串变量，其值为空串
Null	表示空值，用来表示一个变量不包含任何有效数据
Nothing	指示对象变量无关联的实际对象
vbCr	表示回车符
vbCrLf	表示回车/换行符
vbTab	表示制表符

2．变量

VBScript 变量命名要求以字母开头，不能包含句号（.）字符，且长度不能超过 255 个字符。在 VBScript 代码中，可以先声明变量再使用变量（显式声明），也可以不声明变量直接使用（隐式声明）。

显式声明变量使用 Dim 语句，例如：

```
Dim a,b,c        '声明 3 个变量
a=5              '为变量 a 赋值
a=a+1            '将变量 a 的值增加 1
```

在 ASP 代码中可用 Option Explicit 语句强制要求程序中的变量必须显式声明。例如：

```
<% @ Language=VBScript %>
<% Option Explicit %>
<% dim usename
   Dim account_number
%>
```

变量的作用域也称变量的有效范围，由声明它的位置决定。如果在过程中声明变量，则只有该过程中的代码可以访问或更改变量值，此时变量具有局部作用域，并被称为过程级变量。如果在过程之外声明变量，则该变量称为 Script 级变量或全局变量，则同一程序文件的任何脚本代码均可访问或修改该变量的值。

3．数组变量

在成批处理数据时，常用到数组。数组是同一名称的元素的集合，可通过下标区分访问各个元素。VBScript 中的数组具有如下特点：

（1）要先定义，后使用。

（2）下标从 0 开始。

（3）定义数组时可给出数组的上界。

（4）一个数组中可含有不同类型的数组元素。

声明数组变量时变量名后面带有括号()。例如：

```
Dim A(10)
```

表示声明包含 11 个元素的一维数组，这些元素分别是 A(0)、A(1)、……A(10)。

通过下标变量引用数组中某个特定的元素。例如，给下标为 5 的数组元素赋值：

```
A(5) = 10
```

声明多维数组时用逗号分隔括号中表示各维数组上界的数字。例如，以下声明一个 6 行 11 列的二维数组：

```
Dim MyTable(5, 10)
```

另外，也可以声明动态数组，即运行时数组的大小可发生变化。对动态数组的最初声明使用 Dim 语句或 ReDim 语句。但是对于动态数组，则括号中不包含任何数字。例如：

```
Dim MyArray()
ReDim AnotherArray()
```

要使用动态数组，必须随后使用 ReDim 语句确定维数和每一维的大小。在下面的代码中，ReDim 将动态数组的初始大小设置为 25，而后面的 ReDim 语句将数组的大小调整为 30。注意，重新调整动态数组大小的次数是没有任何限制的。

```
ReDim MyArray(25)
ReDim Preserve MyArray(30)
```

其中，使用 Preserve 关键字规定在重新调整大小时保留数组的内容，但应注意，将数组的大小调小时，有可能丢失部分存于数组中的数据。

3.2.3　VBScript 运算符

VBScript 有一套完整的运算符，包括算术运算符、关系运算符、连接运算符和逻辑运算符。其中，连接运算符“&”可实现字符串与任意类型数据的拼接。

1．算术运算符（如表 3-3 所示）

表 3-3　算术运算符

运　算　符	描　　述	运　算　符	描　　述
^	求幂	\	两个数的整除结果
–	负号	Mod	求两个数值相除的余数
*	乘	+	加
/	除	–	减

算术运算符的优先级顺序是：负号→求幂→乘、除→整除→求余→加、减。

其中，乘和除为同一级别，加和减为同一级别。

2．关系运算符（如表 3-4 所示）

表 3-4　关系运算符

运　算　符	描　　述	运　算　符	描　　述
=	等于	>	大于
<>	不等于	<=	小于等于
<	小于	>=	大于等于
Is	判断两个对象引用是否引用同一个对象		

所有关系运算符的优先级相同。

3．逻辑运算符（如表 3-5 所示）

表 3-5　逻辑运算符

运　算　符	描　　述	运　算　符	描　　述
Not	非	Xor	异或
And	与	Eqv	等价
Or	或	Imp	隐含

运算符的优先级顺序是：Not→And→Or→Xor→Eqv→Imp。

4．运算符的优先级

当表达式中包含多个运算符时，将按预定顺序计算每一部分，该顺序被称为运算符优先级。可以使用括号越过这种优先级顺序，运算时，总是先执行括号中的运算符，然后再执行括号外的运算符。但应注意，在括号中仍遵循运算符优先级。

当表达式中包含多种运算符时，首先计算算术运算符，然后计算关系运算符，最后计算逻辑运算符。注意，同一级别优先级按从左到右的顺序计算。

字符串连接运算符（&）不是算术运算符，但是在优先级顺序中，它排在所有算术运算符之后和所有关系运算符之前。例如，以下代码将字符串与数字进行拼接。

```
<script type="text/vbscript">
   x="welcome"& vbCrLf & 2008 & " beijing!"
   MsgBox(x)
</script>
```

在网页的浏览界面中将弹出如图 3-2 所示的对话框。



图 3-2　使用&运算符

需要注意的是，使用加号（+）也可以进行字符串的拼接，但不能用加号将字符串与其他类型的数据拼接。以下表达式为非法表示，不能完成计算。

```
x="welcome"& vbCrLf + 2008 &" beijing!"
```

但如果字符串可转换为数值数据，则可以将字符串与数据进行相加。例如，x="12.4"+1，则 x 的结果为 13.4。

3.3　VBScript 的流程控制语句

按照一般程序的流程结构类型，即顺序结构、分支结构和循环结构 3 种类型，可以将 VBScript 中的流程控制语句分为两类。第一类为分支结构的流程控制语句；第二类为循环结构的流程控制语句。分支结构的流程控制语句包含 if 语句和 Select Case 语句，而循环结构的流程控制语句则有众多情形。

3.3.1　if 语句

1．单行 if 语句

格式：IF　条件表达式 THEN 语句体 1　[ELSE　语句体 2]

功能：如果条件表达式结果为真，则执行 THEN 后的语句体 1，否则执行 ELSE 后的语句体 2，如果无 ELSE 部分，则当条件不满足时这条语句就什么也不做。

例如：

```
if time > #6:00:00pm# then MsgBox("Good Evening! ")
```

这里，利用 time 函数获得当前的时间值，如果时间超过了下午 6 点，则弹出显示 Good Evening 的消息框。

2．块 if 语句

格式：

```
IF 条件表达式 THEN
   语句体 1
[ ELSE
   语句体 2 ]
END IF
```

其中，ELSE 分支可以没有，但 END IF 不可少，因为只有遇到它才代表语句块的结束。

与单行 if 不同的是，语句体 1 和语句体 2 中可放置多条语句。

【例 3-2】　在客户端脚本中使用 if 语句举例

```
-------------------------------------------------------ex3-2.htm-------------------------------------------------------
<script type="text/vbscript">
     vMon = Month(Date)     'Date 函数返回当前的系统日期，Month 求日期的月份
     if vMon >= 3 and vMon <11 then
          MsgBox("太好了")
          document.write("适合出去玩。")
     else
          MsgBox("除非喜欢玩冰")
          document.write("天太冷了，呆在家吧。")
```

```
    end if
</script>
```

【说明】该客户端脚本程序利用 document 对象的 write 方法在页面上输出信息。输出信息的位置取决于脚本代码在页面中的位置。这里用到第 5 章将要介绍的浏览器对象模型，document 代表当前页面文档对象。

【思考】将上述代码改成在服务器端执行的脚本代码，应如何修改程序？

if…then…else 语句的一种变形是从多个条件中选择，即添加一个或多个 elseif 子句以扩充 if…then…else 语句的功能。

【例 3-3】　根据输入的学生成绩输出不同的信息

```
--------------------------------------------------------ex3-3.htm--------------------------------------------------------
<script type="text/vbscript">
     s = cint(InputBox("请输入一个成绩",""))
     if   s<60   then
          document.write( "不及格")
     elseif s<70 then
          document.write("及格")
     elseif s<80 then
          document.write( "中")
     elseif s<90 then
          document.write( "良")
     else
          document.write("优")
     end if
</script>
```

使用多个 elseif 子句会变得很累赘，在多个条件中进行选择的更好方法是使用 Select Case 语句。

3.3.2　Select Case 语句

Select Case 结构可以从多个语句块中选择执行其中的一个，该结构在其开始处使用一个只计算一次的简单测试表达式。表达式的结果将与结构中每个 Case 的值比较。如果匹配，则执行与该 Case 关联的语句块，语句格式如下：

```
SELECT CASE 测试表达式
  CASE 表达式 1
    语句体 1
  CASE 表达式 2
    语句体 2
  ……
  CASE 表达式 n
    语句体 n
```

```
    CASE ELSE
        语句体
END SELECT
```

【例 3-4】　Select Case 语句的使用

```
----------------------------------------------------------ex3-4.htm----------------------------------------------------------
<script type="text/vbscript">
    vDay = WeekDay(Date)
    Select Case vDay
      Case 1
        document.write("今天是星期天。")
      Case 2
        document.write("今天是星期一。")
      Case 3
        document.write("今天是星期二。")
      Case 4
        document.write("今天是星期三。")
      Case 5
        document.write("今天是星期四。")
      Case 6
        document.write("今天是星期五。")
      Case else
        document.write("今天是星期六。")
    end select
</script>
```

【说明】这里用 Date 函数获得当前日期，用 WeekDay 函数计算当前日期是星期几。

【注意】Select Case 结构只计算开始处的一个表达式（只计算一次），而 if…then…elseif 结构计算每个 elseif 语句的表达式，这些表达式可以各不相同。仅当每个 elseif 语句计算的表达式都相同时，才可以使用 Select Case 结构代替 if…then…elseif 结构。

3.3.3　循环语句

在 VBScript 中可使用下列循环语句。

- ❑ Do…Loop：当（或直到）条件为 True 时循环。
- ❑ While…Wend：当条件为 True 时循环。
- ❑ For…Next：指定循环次数，使用计数器重复运行语句。
- ❑ For Each…Next：对于集合中的每项或数组中的每个元素，重复执行一组语句。

1．Do…Loop 循环

Do 循环一般用于事先不知道循环次数的循环，可分为使用 While 检查条件和 Until 检查条件两种情形。

（1）使用 While 检查条件

只要条件为 True，就会进行循环。有以下两种形式：

① 将条件检查放在循环开始（Do While…Loop）。

② 将条件检查放在循环末尾（Do…Loop While）。

以下为 Do While…Loop 循环的客户端脚本代码示例：

```
counter = 0
myNum = 20
Do While myNum > 10
    myNum = myNum - 1
    counter = counter + 1
Loop
MsgBox "循环重复了 " & counter & " 次。"
```

以下为 Do…Loop While 循环的客户端脚本代码示例：

```
n=1
fac=1
Do
    n=n+1
    fac= fac * n
Loop While n < 10
MsgBox n & "!=" & fac
```

【思考】以上两个程序输出的结果分别如何？

（2）使用 Until 检查条件

只要条件为 False，就会进行循环。有以下两种形式：

① 将条件检查放在循环开始（Do Until…Loop）。

② 将条件检查放在循环末尾（Do…Loop Until）。

以下为 Do Until…Loop 循环的客户端脚本代码示例：

```
counter = 0
num = 10
Do Until num = 5
    num = num - 1
    counter = counter + 1
Loop
MsgBox "循环重复了 " & counter & " 次。"
```

【注意】在 Do 循环内可用 Exit Do 语句在特定条件下退出循环。

2．While…Wend 循环

While…Wend 语句是从早期的 Basic 语言中保留下来的一种循环，语法格式如下：

While 条件

循环体

Wend

功能等价于如下形式的 do while 循环：

do while 条件
　循环体
loop

3．For…Next 循环

For…Next 语句用于可事先计算循环次数的循环。循环控制变量的值由初值变化到终值为止，随每一次循环按步长增加或减少。其语法格式如下：

for 变量=初值 to 终值[step 步长]
　语句体
next

其中，语句体中可包括 exit for 语句，用来在特定条件下退出循环。

要使控制变量的值递减变化，可将步长设为负值。此时终值必须小于初值。在下面的示例中，变量 n 每次减 2。循环结束后，total 的值为 16、14、12 和 10 的累加和。

```
For n = 16 To 10 Step -2
    total = total + n
Next
```

4．For Each…Next 循环

For Each…Next 循环用于遍历访问集合或数组中的元素。例如，以下 ASP 代码使用了两种循环访问数组中的元素，前面 for 循环为数组赋值，后面 for each 循环输出所有数组元素的值。

```
<%
  dim x(10)
  for k=0 to 9
      x(k)= k+1
  next
  for each m in x
      response.write m & " "
  next
%>
```

3.4　VBScript 的过程定义与调用

在 VBScript 中，可以将完成特定功能的一段程序定义为过程，按照是否有返回值可将过程分为两类，即 Sub 过程和 Function 过程。

3.4.1　Sub 过程及其调用

Sub 过程是无返回值的过程，一般格式如下：

Sub 过程名(形式参数列表)

　语句块

　[exit sub]

End Sub

其中，形式参数列表中列出了所有的参数，各参数之间用逗号分隔。如果过程定义的小括号内未列出任何参数，则该过程称为无参过程。过程体中的 exit sub 语句用于在特定条件下退出过程。

下面是一个带有两个参数的 Sub 过程，其功能是用消息框显示两个参数的乘积值。

```
Sub myMulti(no1, no2)
    MsgBox(no1*no2)
End Sub
```

调用 Sub 过程有两种方法：一种是直接通过输入过程名后跟相应的实参；另一种是用 Call 语句调用，注意此时必须将所有参数包含在括号之中。例如：

```
Call myMulti(8,9)
myMulti 8,9
```

【注意】当不使用 Call 语句调用时，不加括号；当使用 Call 语句时，要用括号包含所有参数。

3.4.2　Function 过程及其调用

Function 过程也称为函数，通过函数名返回一个值，在函数体内必须给函数名赋值。函数的语法格式如下：

Function 函数名(形式参数)

　语句块

　[exit function]

　函数名=函数的返回值

End Function

其中，exit function 语句用于在特定条件下结束函数的执行。

以下 Function 过程的返回值是两个参数中的最大数。

```
Function max(no1, no2)
  if no1> no2 then max= no1   else max= no2
End Function
```

函数调用可出现在表达式中，当调用 Function 过程时，必须为函数传递实际参数。

例如，x = max(50, 35)+10，则 x 的结果为 60。

3.5　VBScript 中的内部函数

3.5.1　转换函数

通过转换函数将子数据类型转换成需要的数据类型，常用的转换函数如下。

（1）CDate(Variant)：转换成日期子类型。

（2）CStr(Variant)：转换成字符串子类型。

（3）CByte(Variant)：转换成 Byte 子类型。

（4）CInt(Variant)：转换成整数子类型。

（5）CBool(Variant)：转换成布尔子类型。

另外，还有两个经常使用的函数是字符与 ASCII 码的转换函数。

（1）chr(int)：求 ASCII 码对应的字符。

（2）asc(string)：求字符串中第一个字符的 ASCII 码。

3.5.2　字符串函数

在 VBScript 中包含有很多处理字符串的函数，常用的字符串函数如下。

（1）Len(string)：返回字符串 string 中的字符数目。

（2）Trim(string)：将字符串 string 前后的空格去掉。

（3）Mid(string,start,length)：从字符串 string 的 start 字符开始取得 length 长度的字符串，如果省略第三个参数，则表示是取从 start 字符开始到字符串结尾的字符串。

（4）Left(string,length)：从字符串 string 的左边取得 length 长度的字符串。

（5）Right(string,length)：从字符串 string 的右边取得 length 长度的字符串。

（6）LCase(string)：将字符串 string 中的所有大写字母转化为小写字母。

（7）UCase(string)：将字符串 string 中的所有小写字母转化为大写字母。

（8）StrComp(string1,string2)：返回 string1 字符串与 string2 字符串的比较结果，如果两个字符串相同，则返回 0。若 string1 大于 string2，则返回正数，否则返回负数。

（9）replace(string,old,new)：将字符串 string 中的所有 old 子串用 new 串替换。

（10）instr(string,substr)：查找字符串 string 中 substr 子串首次出现的位置。如果没有 substr 子串，则返回 0。

（11）instr(pos,string,substr)：从字符串 string 中的 pos 位置开始往后查找 substr 子串首次出现的位置。

（12）split(string,substr)：该函数返回一个字符串数组，数组元素是使用 substr 子串作为分隔符对 string 串中的内容进行切分后得到的所有子串。

例如，split("1-0-0,2-0-0,4-0-0",",")得到的数组包括 3 个元素，它们的值分别为"1-0-0"、

"2-0-0"和"4-0-0"。

【思考】用“-”切分得到的结果又如何？

【例 3-5】　对显示文本进行变换处理的自定义函数

```
<%
Function convert(mytext)
    mytext = replace(mytext, "'", "’ ")      '将英文单引号用中文单引号替换
    mytext = replace(mytext, chr(13) & chr(10), "<br>")
    mytext = replace(mytext, chr(10), "<br>")
    mytext = replace(mytext, chr(13), "<br>")
    mytext = replace(mytext, chr (32)," ")
    convert = mytext
End Function
%>
```

【说明】这里用 replace 方法对文本中的单引号、回车/换行符及空格进行替换处理，从而使得在浏览器显示内容时能保持文本的原来风格。防止拼接 SQL 查询语句时因为英文单引号的出现导致语句格式错误。

3.5.3　日期和时间函数

在 VBScript 中，可以使用日期和时间函数来得到各种样式的日期和时间。常用的日期和时间函数如下。

（1）Now()：返回系统当前的日期和时间。

（2）Date()：返回系统当前日期。

（3）Time()：返回系统当前时间。

（4）Year(Date)：返回给定日期的年份。

（5）Month(Date)：返回给定日期的月份。

（6）Day(Date)：求给定日期是几号。

（7）Hour(time)：求给定时间的小时值。

（8）Minute(time)：求给定时间的分钟值。

（9）Second(time)：求给定时间的秒值。

（10）WeekDay(Date)：返回日期是星期几的整数，1 表示星期日，2 表示星期一，依此类推。

（11）DateDiff("Var",Var1,Var2)：计算两个日期或时间的间隔，其中，Var 表示间隔因子，Var1 表示第一个日期或时间，Var2 表示第二个日期或时间。

（12）DateAdd("Var",Var1,Var2)：对两个日期或时间作加法。

3.5.4　数学函数

VBScript 中常用的数学函数如下。

（1）Abs(number)：返回一个数的绝对值。

（2）Sqr(number)：返回一个数的平方根。

（3）Sin(number)：返回角度的正弦值。

（4）Cos(number)：返回角度的余弦值。

（5）Tan(number)：返回角度的正切值。

（6）Log(number)：返回一个数的对数值。

（7）Int(number)：取整函数，返回小于等于 number 的第一个整数。

（8）Rnd(number)：产生 0～1 之间的随机数。

（9）Ubound(数组名称, 维数)：返回该数组的最大下标数。如果数组只有一维，则可以省略第二个参数维数。

【说明】用 Rnd 函数产生随机数时，一般要先用 Randomize 对序列初始化，否则每次运行程序产生的随机序列都是一个不变的序列。

【例 3-6】　产生一个在一定范围（0～range）内不含重复数的随机序列

以下函数包含两个参数：range 为抽数的最大值，total 为要产生的随机数的个数。程序中将抽到的数存入一个数组（shiti）中和一个字符串（havepick）中。为方便查找处理，处理过程中将已抽数登记在字符串 numstr 中，其中的每个数添加了前缀“-”和后缀“,”。

```
<%
Function genrandom(total, range)
reDim shiti(total)
Randomize                                       '对随机序列初始化
havepick=""                                     '存放抽到的数
pickamount=0                                    '已抽取数的个数
do while pickamount < total
    num = int(Rnd(1)* range )+1
    numstr = "-"&num&","
    if   instr(havepick,numstr) = 0 then
        shiti(pickamount) = num                 '不能存放字符串，否则排序不正确
        havepick = havepick & numstr            '登记已抽取的数
        pickamount = pickamount +1
    end if
loop
'以下将序列中的数据按由小到大排序
For ai = 0 To pickamount-1
     For aj = ai + 1 To pickamount-1
        If shiti(ai)> shiti(aj) Then
            temp = shiti(ai)
            shiti(ai) = shiti(aj)
            shiti(aj) = temp
         End If
     Next
Next
'以下将数值数据拼接为字符串，各数之间用逗号隔开
havepick= shiti(0)
For ai = 1 To   pickamount-1
```

```
    havepick = havepick &"," & shiti(ai)
next
genrandom = havepick
end function
%>
```

【应用思考】该函数可用于考试系统中的随机抽题，序列的数值代表记录集的记录号。

3.5.5　检验函数

检验函数通常用来检验某变量是否是某种类型，常用的检验函数如下。

（1）VarType(Variant)：函数值为该变量的数据类型，0 表示空（Empty），2 表示整数，7 表示日期，8 表示字符串，11 表示布尔变量，8129 表示数组。

（2）IsNumeric(Variant)：如果 Variant 是数字类型，则函数值为 True。

（3）IsDate(Variant)：如果 Variant 是日期类型，则函数值为 True。

（4）IsNull(Variant)：如果 Variant 是无效值，则函数值为 True。

（5）IsEmpty(Variant)：如果 Variant 没有设定值，则函数值为 True。

（6）IsObject(Variant)：如果 Variant 是对象类型，则函数值为 True。

（7）IsArray(Variant)：如果 Variant 是数组类型，则函数值为 True。

3.5.6　输入与输出函数

1．输入函数 InputBox

InputBox 函数的作用是显示一个输入对话框，用于用户输入信息。其语法格式如下：

InputBox(prompt [,title][,default])

【注意】“[”和“]”中的内容表示可选项。

其中：

（1）prompt：显示在对话框中的文字。

（2）title：显示在对话框标题中的文字。

（3）default：指定对话框中输入框内的默认值。

函数返回值为用户从键盘输入的字符串内容。

2．输出函数 MsgBox

MsgBox 函数的作用是弹出一个消息框，用于告知用户信息。其语法格式如下：

MsgBox(prompt [,button][,title])

其中：

（1）prompt：显示在消息框中的文字。

（2）button：是一个数值表达式，用于指定显示按钮的数目及形式，使用图标样式等。Button 参数的常用设置值如表 3-6 所示。具体值可以是几类常量的相加组合。

（3）title：显示在消息框标题中的文字。

表 3-6　button 参数的常用设置值

常　　量	值	含　　义
vbOKOnly	0	只显示 OK 按钮
vbOKCanel	1	只显示 OK 及 Canel 按钮
vbAbortRetryIgnore	2	只显示 Abort、Retry 及 Ignore 按钮
vbYesNoCanel	3	只显示 Yes、No 及 Canel 按钮
vbYesNo	4	只显示 Yes 及 No 按钮
vbRetryCanel	5	只显示 Retry 及 Canel 按钮
vbCritical	16	显示重要消息图标
vbQuestion	32	显示询问图标
vbExclamation	48	显示警告消息图标
vbInformation	64	显示信息消息图标
vbDefaultButton1	0	第 1 个按钮是默认值
vbDefaultButton2	256	第 2 个按钮是默认值
vbDefaultButton3	512	第 3 个按钮是默认值
vbDefaultButton4	768	第 4 个按钮是默认值

【例 3-7】　判断一个输入数是否为素数，若是，则输出 yes，否则输出 no

```
---------------------------------------------------------ex3-7.htm---------------------------------------------------------
<script type="text/vbscript">
n = cint(InputBox("请输入一个整数"))
k=2
do while k<n
  if n mod k=0 then exit do
  k=k+1
loop
if k<n then
   MsgBox " no "
else
   MsgBox "yes "
End if
</script>
```

【说明】这里使用了 InputBox 函数读取用户从键盘输入的数据，并用 cint 函数将数据转换为整数。注意，根据素数的定义条件组织循环，在循环中只要有一个数能除尽 x 就可以判断不是素数，利用 exit do 退出循环。

【注意】输入与输出函数只能在客户端运行，而不能在服务器端运行。

本 章 小 结

VBScript 是 ASP 脚本编程的默认语言，它既可用于服务器端的 ASP 脚本编程，又可

用于客户端的事件驱动编程。由于除了 IE 浏览器外，其他浏览器不支持 VBScript，因此，在实际应用中它主要用于服务器端的 ASP 编程。本章介绍 VBScript 的数据类型和语言基本组成。读者在熟悉语言的基本语句和常用函数使用的基础上，注意把握 VBScript 的数据类型的特点，熟悉数组的定义和使用，掌握过程与函数的定义与调用方法。另外，VBScript 还有一个复杂的内容就是客户端的事件处理编程，本书中不进行介绍，客户端编程建议用户尽量使用 JavaScript，因为所有浏览器均支持 JavaScript 代码。

习　　题

1．选择题

（1）以下说法正确的是（　　）。

A．VBScript 的注释符号为“//”

B．常量是一种不会改变的数，定义常量要使用“Dim 常量名称”

C．运算符“+”也可以用来连接字符串

D．运算符 Mod 可以计算出除法结果中的整数

（2）关于 VBScript 的命名规则，下面说法中不正确的是（　　）。

A．第一个字符必须是数字或字母

B．长度不能超过 255 个字符

C．名字不能和关键字同名

D．在声明时不能声明两次

（3）使用（　　）语句可以立即从 Sub 过程中退出。

A．Exit Sub　　B．Exit

C．</Sub>　　D．Loop

（4）判断程序运行完毕后，x、y、z 的值分别为（　　）。

```
x = "11" + 1
y = "11" & 1
z = "11" + "1"
```

A．111,111,111　　B．12,111,12

C．12,111,111　　D．12,12,12

（5）下列（　　）函数可以将数值型转换为字符串。

A．CDate　　B．CInt　　C．CStr　　D．CDbl

（6）函数 mid("1234567890", 3, 3)的返回值是（　　）。

A．345　　B．234　　C．456　　D．7890

（7）在 VBScript 中，注释采用（　　）。

A．//　　B．/*…*/　　C．’　　D．”

（8）若要求 VBScript 的变量在使用前必须事先定义，则应使用（　　）语句来设置。

A．Option Explicit　　B．Option Base
C．Dim　　D．ReDim

（9）函数 Instr("xxPPppXXpx","pp")的返回值为（　　）。

A．3　　B．5　　C．2　　D．4

（10）退出 Do 循环，应使用的语句是（　　）。

A．Exit　　B．Exit Sub　　C．Exit For　　D．Exit Do

2．问答题

（1）写出下面几个表达式的运算结果。

① 19 Mod 5 * 3　　② −3^2　　③ "3.1415" & 926　　④ 234 & 0.12345
⑤ "50" + "50"　　⑥ "x" > "y"　　⑦ (9>3)Xor("x">"y")

（2）VBScript 中变量的数据类型是什么？有什么特点？

（3）VBScript 中的变量和数组在功能上有何异同？

3．编程题

（1）试编写程序段，将字符串“美丽”和“中华”两边的空格去掉，并连成一个字符串“美丽中华”。

（2）编写函数 f(x,y)计算 x 和 y 的平方和，并调用该函数求 2 和 6 的平方和。

（3）$S=1^2+3^2+5^2+\cdots+199^2$，试利用两种循环语句编写程序计算 S 的值。

（4）设有一个字符串表示一个知识点集合，前后为小括号，知识点之间用逗号分割，统计其中知识点的数量，分离出该字符串中的每个知识点，如(1-0-0,2-1-0,4-0-0)。

（5）编写一个程序，在客户端弹出对话框让用户输入一个整数 n，计算 n 的阶乘，并将结果通过提示框显示。

（6）编写一个程序，在服务器端利用随机函数产生一个小于 10 的整数 n，计算 n 的阶乘，并在网页中显示 n!=xxx 的输出。

（7）编程实现在页面中输出 100 个随机键位的大写字母，要求每行显示 20 个字母，各字母间用一个空格分割。

方法 1：定义 100 个数组元素的数组，用来存放产生的随机字母。参考代码如下：

```
<%
dim ran(100)
randomize
for i=1 to 100
   ran(i) = chr(int(26*rnd())+65)
next
for i=1 to 100
   Response.Write(ran(i) & "   ")
   if i mod 20=0   then
        Response.Write("<br>")
   end if
next
%>
```

方法 2：定义用于存放 26 个随机字母的数组，通过产生 100 个随机获取数组元素的下标地址来获得这些字母。

（8）编写求阶乘的函数，并利用该函数实现求组合的函数。

n 个元素中取 m 个元素的组合为 C(n,m)=n!/((n-m)!*m!)。

利用求组合方法输出如下杨辉三角形。

c(0,0)

c(1,0) c(1,1)

c(2,0) c(2,1) c(2,2)

c(3,0) c(3,1) c(3,2) c(3,3)

要求分别在客户端代码和服务器端代码中实现输出。

第 4 章　ASP 的内置对象

ASP 提供了 6 个内置对象，这些对象在使用时不需要任何声明和建立的过程，可在代码中直接调用它们，其功能如表 4-1 所示。

表 4-1　ASP 的内置对象及功能

对 象 名 称	功　　能
Request 对象	用于获取 HTTP 请求传递的所有信息，包括从 HTML 表单用 POST 或 GET 方法传递的信息、Cookie 等
Response 对象	用于发送信息给客户端、重定向浏览器到另一个 URL 和设置 Cookie 的值等
Server 对象	提供了一系列的方法和属性，为应用程序提供特定服务，最常用的是 Server.CreateObject 方法，它允许在服务器上创建对象
Session 对象	用于存储用户的个性化信息，其保存的用户状态信息在 Session 过期或删除前一直有效
Application 对象	用来存储和获取可以被该应用程序所有用户共享的信息
ObjectContext 对象	用于提交或中止一个由 Microsoft 事务服务器（MTS）管理的事务，通过 ASP 脚本对事务服务进行初始化

在这些对象中，最常用的是 Request 对象和 Response 对象，它们实现了客户浏览器与 Web 服务器的交互功能。Request 对象对应 HTTP 请求，而 Response 对象对应 HTTP 响应。

4.1　Request 对象

Request 对象用于访问客户端用 HTTP 请求传递的信息，将客户端数据保存到内置的几个集合中，如表 4-2 所示。通过访问这些集合，可分别获得表单所提交的数据、Cookie 的值以及服务器环境的值等。Request 对象只有一个属性 TotalBytes，其功能是取得客户端请求数据的字节大小。同时 Request 对象只有一个方法 BinaryRead，其功能是读取通过 POST 请求从客户端传送到服务器的二进制数据。

表 4-2　Request 对象的集合

集 合 名 称	功能与用途
Form	获得用 POST 方法提交的表单数据
QueryString	获得用 GET 方法提交的表单数据
Cookies	获得 Cookie 的值
ServerVariables	获得 HTTP 头文件的相关信息和服务器环境变量的值
ClientCertificate	获得 HTTP 请求的客户端证书中的字段值

Request 对象集合的数据访问方式如下：

Request.集合名(参数)

其中，参数可以是字符串形式表示的一个元素名或数值形式表示的元素序号。

【注意】也可以省略集合名，而直接通过 Request(参数)访问数据集合的成员，这时 ASP 会按照 QueryString、Form、Cookies、ClientCertificate、ServerVariables 的顺序在各个数据集合中搜索该变量，并返回第一个出现的变量值。显然，如果不指定数据集合，则一方面会影响执行效率，另一方面还可能由于多个集合中有同名元素而出现访问歧义性。

4.1.1　Form 集合

1．基本使用方式

Form 集合是使用 POST 方法提交的表单数据。其语法格式如下：

Request.Form(参数)[(索引)| .Count]

其中：

- 参数：用来指定表单元素名或元素序号。集合中元素序号是从 0 开始的。
- 索引：为可选项，使用该选项可以访问某参数中多个值中的一个。它可以是 1 到 Requset.Form(参数).Count 之间的任意整数。
- Count：表示集合中具有相同参数名的元素的个数。

以下为表单访问的常用形式举例。

（1）Request.Form("username")：访问表单中名为 username 的域。

（2）Request.Form("hobby").count：假设 hobby 为一个多值元素，则该式求元素的个数，而 Request.Form("hobby")的结果是一个集合。

（3）Request.Form("hobby")(i)：求 hobby 的序号为 i 的值，i 从 0 开始。

（4）Request.Form.count：表单中的元素个数。

（5）Request.Form(i)：访问表单中序号为 i 的域，i 从 0 开始。

【例 4-1】　简单注册提交程序

以下注册程序将用户填写的注册信息通过注册处理程序获取后显示在页面中。

程序 1：注册界面

```
------------------------------------------------------regist.html------------------------------------------------------
<html>
<body>
<center>用户注册
<form action="regist_process.asp" method="post" name="myform">
<table border="0" bordercolor="#FF0000" width=50%>
<tr><td width="40%" >用户名：</td>
<td width="60%" ><input name="username" type="text"></td></tr>
<tr><td>密码：</td><td><input name="userkey" type="password" ></td></tr>
<tr><td>性别：</td><td >
<input name="sex" type="radio" value="男" checked>男
```

```
<input type="radio" name="sex" value="女">女
</td></tr>
<tr><td >教育水平：</td><td>
<select name="edu" size="1" id="edu">
<option value="硕士以上">硕士以上</option>
<option value="大学本科">大学本科</option>
<option value="大专">大专</option>
<option value="中专以下">中专以下</option>
</select>
</td></tr>
<tr><td >电子邮箱：</td><td ><input name="email" type="text" ></td>
</tr><tr><td> </td><td> </td></tr>
<tr><td colspan="2" >
<div align="center">
<input type="submit" name="register" value=" 注册 " >
<input type="reset" name="input_again" value="全部重写">
</div>
</td></tr>
</table>
</form>
</center>
</body>
</html>
```

运行结果如图 4-1 所示。



图 4-1　用户注册

程序 2：注册处理

```
--------------------------------------------------regist_process.asp--------------------------------------------------
<html>
<body>
注册的用户名为：<%=request.form("username")%><br>
输入密码为：<%=request.form("userkey")%><br>
性别为：<%=request.form("sex")%><br>
教育程度为：<%=request.form("edu")%><br>
电子邮箱为：<%=request.form("email")%>
```

```
</body>
</html>
```

运行结果如图 4-2 所示。

【说明】注意两个程序之间的关系，regist.html 用于实现注册信息的填写界面，其中包含有一个表单，表单的 action 属性规定了表单提交后将传给 regist_process.asp 程序进行处理。在 regist_process.asp 程序中可通过 request 对象读取表单控件提交的数据。

2．遍历表单元素

如果要遍历输出表单提交的所有数据，可以利用 for 循环实现。

方法 1：利用元素与集合的从属关系通过 for each 循环访问集合元素。

```
<%
for each item in request.form
      response.write "<br>"&item&":"&request.form(item)
next
%>
```

方法 2：利用序号访问集合元素。

```
<%
for i=1 to request.form.count
      response.write(request.form(i)&"<br>")
next
%>
```

【例 4-2】　获取多项选择题的用户解答

以下为网络教学中演示多选题做题界面的应用。在输入表单中，利用循环产生 5 个名字均为 choice1 的复选框，它们的值分别为 A、B、C、D、E。

文件 1：试题及解答表单显示页面（dxzt.asp）

```
------------------------------------------------------------dxzt.asp----------------------------------------------------------
<!--…… 试题内容显示部分涉及访问数据库，略 -->
<form name="my" METHOD="post" ACTION="process.asp" >
<table width=80%>
<%
str="ABCDE"
for k=1 to 5
%>
<td align=left>
<input type="checkbox"  name="choice1"  value="<%=mid(str,k,1)%>" ><%= mid(str,k,1)%>
</td>
<% next %>
</table>
<P align=center><input type="submit" value=" 提交 "></p>
</form>
```

运行结果如图 4-3 所示。



图 4-2　显示用户填写的信息

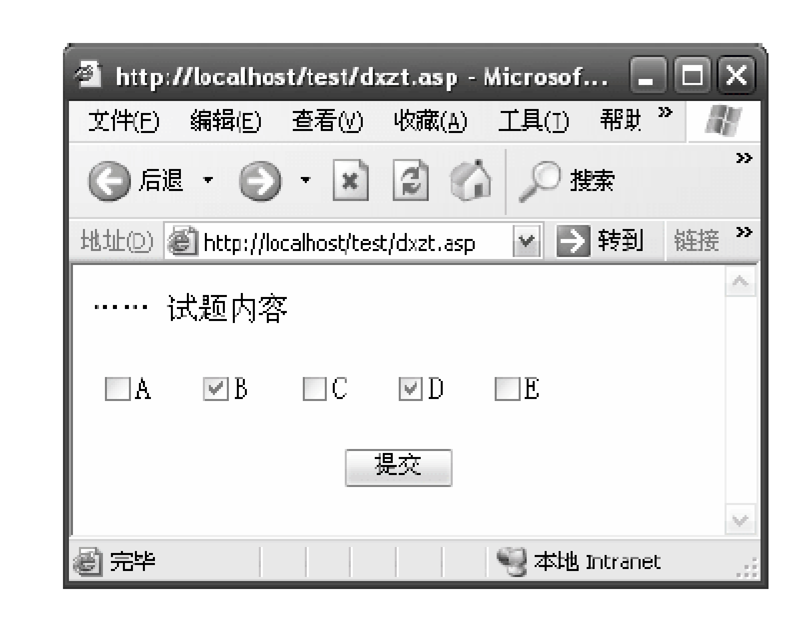


图 4-3　多选题解答界面

文件 2：判定解答处理程序

在网络教学系统中，该程序实际上是将用户的解答与数据库中的标准答案进行比较，根据比较结果给出不同的响应信息。这里不妨仅将用户提交的解答输出。

```
------------------------------------------------------process.asp------------------------------------------------------
<html>
<body>
-----你选择的答案是：<%= request.Form("choice1") %>
</body>
</html>
```

对应图 4-3 中显示的用户操作，解答处理程序的输出结果为：

-----你选择的答案是：B，D

可以看出，结果为一个数据集合，只有被选中的选项才会出现在多值集合中。要遍历集合中的所有元素，可通过 for 循环实现。以下为 ASP 程序代码：

```
answer=""
Response.Write "-----你选择的答案是："
for n=1 to request.Form("choice1").count
    answer=answer & request.Form("choice1")(n-1)
next
Response.Write(answer)
```

如此以来，显示的输出结果为：

-----你选择的答案是：BD

如果用 for each 循环访问以上集合元素，可写成如下形式：

```
for each item in request.Form("choice1")
     answer=answer & item
next
```

4.1.2　QueryString 集合

QueryString 集合用于检索 HTTP 查询字符串中变量的值，HTTP 查询字符串由 URL 中问号（?）后的内容指定。例如：

```
<A HREF= "example.asp?info=this is a sample">string sample</A>
```

表示该超链接的 URL 中有一个参数名为 info，值为"this is a sample"。如果有多个参数，则参数间用“&”分隔。

利用 QueryString 集合获取客户端数据的语法格式如下：

Request.QueryString(variable)[(index)|.Count]

例如：

```
request.querystring("info")
```

一般地，QueryString 集合用于获取超链接传递的信息，例如：

```
<html>
<body>
<p>选择课程：<br>
<a href="enterkc.asp?kc=java">Java 语言</a><p>
<a href="enterkc.asp?kc=web">Web 编程技术</a><p>
</body>
</html>
```

【说明】页面中有多个目标为 enterkc.asp 的超链接，每个超链接含有名为 kc 的 URL 参数。当用户单击超链接时，在 enterkc.asp 程序中可以通过如下形式知道用户选择了哪门课程。

```
Request.QueryString("kc")
```

【应用举例】在网上教学系统的首页可通过这样的形式让用户选择进入学习的课程。

如果表单提交方式使用的是 get 形式，则其数据域将附加在处理程序的 URL 地址后面，如 http://localhost/login.asp? username=guest&password=123456。

这样就类似于超链接的访问形式。如果是单帧的页面，则在地址栏中将显示一个长串的 URL，表单输入均暴露在 URL 参数中，包括输入的密码也是明文显示，这就显得有些不合适。另外，get 方法对传输的数据量受到 IE 地址栏长度的限制的影响，不允许超过 2K。因此，表单提交一般采用 post 方式。

4.1.3　Cookies 集合

通过表单和 URL 查询参数时只能在相邻的页面间传递数据信息，在实际应用系统中经常需要持久保持客户的状态信息。例如，客户登录后的身份在应用系统中只要登录一次，则在后续页面中将始终知道用户的身份。由于 HTTP 协议的无状态性，浏览器与服务器的连接只维持在一次请求响应动作过程中，必须设法在客户端记住客户与网站的关联信息。Cookie 即是存储在客户端硬盘上的一种变量，常用于保存用户的状态信息。Cookie 保存的数据信息将随每次 HTTP 请求传递给服务器。Request 对象的 Cookies 集合用于检索用户在 HTTP 请求中发送的 Cookie 值。

需要注意的是，出于安全方面的考虑，有的客户可能会禁用自己计算机上浏览器的 Cookie，这时就不能正常运行含有 Cookie 访问的应用。

读取 Cookies 集合的语法格式如下：

Request.Cookies("Cookiename")[(Key)|.attribute]

其中：

- Cookiename 为 Cookie 变量的名称，用于指定要检索其值的 Cookie。
- Key 代表子关键字的名字，当 Cookie 含有子关键字时，用于检索子关键字的值。
- attribute 为 Cookie 的属性，用于指定 Cookie 自身的有关信息，如 HasKeys 属性用来检查 Cookie 是否包含有子关键字。

和前面集合的访问类似，对集合元素的访问可以通过元素名称，也可以通过元素序号，建议通过元素名称访问，这样更清楚，也不容易引起误会。如果一个 Cookie 包含有子关键字，则访问 Cookie 元素通常要指定子关键字。如果访问 Cookie 时未指定子关键字，则所有关键字的内容将作为一个查询字符串返回。

例如，如果 MyCookie 有 First 和 Second 两个关键字，则当获取 Cookie 值时，如果不指定关键字，代码如下：

```
<%= Request.Cookies("MyCookie")%>
```

那么将输出全部关键字的信息。结果为 First=firstkeyvalue&Second=secondkeyvalue。

如果指定了关键字，则可得到相应关键字的信息，例如：

```
<%= Request.Cookies("MyCookie")("First")%>
```

结果为关键字 First 的值。

要确定某个 Cookie 是否有子关键字，可使用 HasKeys 属性进行检测。例如：

```
<%= Request.Cookies("myCookie").HasKeys %>
```

【应用举例】在用户登录页面处理程序中，如果检查用户合法，则将用户标识通过如下语句存储在 Cookie 变量 username 中：

```
response.cookies("username")=request.form("userid")
```

在后续页面中，可以通过读取 Cookie 变量 username 的值获取用户身份，例如：

```
user = request.cookies("username")
```

4.1.4 ServerVariables 集合

当客户端浏览器向服务器发送页面请求时，除了将所请求页面的 URL 地址传送给服务器之外，也将浏览器的类型、版本等信息一起传送给服务器，这些信息统称为请求标头。当服务器响应客户端浏览器的请求时，除了将所请求的文件传递给客户端之外，也将该文件的大小、日期等信息一起传送给客户端，这些信息统称为响应标头。请求标头和响应标

头统称为 HTTP 标头。使用 Request 对象的 ServerVariables 集合可以检索预定的环境变量和 HTTP 标头信息，语法格式如下：

Request.ServerVariables("环境变量名称")

常用的服务器环境变量如表 4-3 所示。

表 4-3　常用的服务器环境变量

环境变量名	内 容 含 义
ALL_HTTP	客户端发送的所有 HTTP 标题文件
CONTENT_LENGTH	客户端发出内容的长度
CONTENT_TYPE	内容的数据类型，如 text/html
LOCAL_ADDR	接受请求的服务器地址
LOGON_USER	用户登录 Windows NT 的账号
QUERY_STRING	查询 HTTP 请求中问号（?）后的信息
REMOTE_ADDR	发出请求的客户机的 IP 地址
REMOTE_HOST	发出请求的客户机的名称
REQUEST_METHOD	用于构造访问请求的 HTTP 方法，如 GET、HEAD、POST 等
SERVER_NAME	服务器的名称
SERVER_PORT	发送请求的端口号
SCRIPT_NAME	正在执行脚本文件的虚拟路径和名称
PATH_TRANSLATED	所请求文件的物理路径
PATH_INFO	URL 请求的路径信息
HTTP_USER_AGENT	浏览器的类型

以下程序给出了访问 CAI 服务器对若干环境变量的测试结果：

```
---------------------------------------------------------testV.asp---------------------------------------------------------
<%
Response.Write Request.ServerVariables("LOCAL_ADDR")&"<br>"
Response.Write Request.ServerVariables("REMOTE_ADDR")&"<br>"
Response.Write Request.ServerVariables("PATH_INFO")&"<br>"
Response.Write Request.ServerVariables("PATH_TRANSLATED")&"<br>"
Response.Write Request.ServerVariables("SCRIPT_NAME")
%>
```

则访问 http://cai.ecjtu.jx.cn/java/testV.asp 的输出结果如下：

172.16.16.166

172.16.16.113

/java/testV.asp

E:\cai\java\testV.asp

/java/testV.asp

使用以下脚本代码可输出服务器全部环境变量。

```
<TABLE>
<TR><TD><B>环境变量</B></TD><TD><B>值</B></TD></TR>
```

```
<% For Each name In Request.ServerVariables %>
<TR>
<TD> <%= name %> </TD>
<TD> <%= Request.ServerVariables(name) %></TD>
</TR>
<% Next %>
</TABLE>
```

4.2　Response 对象

Response 对象与一个 HTTP 响应相对应，它主要用于控制发送给用户的信息，包括发送响应信息给浏览器、重定向浏览器到另一个 URL、设置 Cookie 的值等。

Response 对象的语法格式如下：

Response.数据集合|属性|方法

其中，数据集合、属性和方法这 3 个参数只能选择一个。

4.2.1　Response 对象的属性

1．Charset 属性

Charset 属性将字符集名称附加到 Response 对象中 content-type 标题的后面，用来设置 Web 服务器发送给客户端响应内容的字符编码。例如：

```
<% Response.Charset="gb2312" %>
```

2．ContentType 属性

ContentType 属性用于指定服务器响应的 HTTP 内容类型。如果未指定 ContentType，则默认为 text/HTML。如果发送为普通文本，则 ContentType 属性设置为 text/plain；如果发送为 xml 文本，则 ContentType 属性设置为 text/xml。例如：

```
<% Response.ContentType="text/xml" %>
```

3．Buffer 属性

如果 Buffer 属性为 True，则服务器发送的响应信息写入缓冲区，只有当前页的所有服务器脚本处理完毕或者调用了 Flush 或 End 方法才将缓冲区内容发送给客户端；如果 Buffer 属性为 False，则在服务器处理脚本时，顺序将响应信息直接发送给客户端。

【注意】

（1）服务器将输出发送给客户端浏览器后，不能再设置 Buffer 属性，因此，通常在 ASP 文件的第一行设置 Response.Buffer 的值。

（2）在 IIS5.0 以后版中，Buffer 属性默认值为 True，而在以前版中，Buffer 属性默认

值为 False，或者用户也可以在 IIS 控制台进行设置。设置缓冲的目的是提高应用的整体效率，但如果 ASP 代码执行时间过长，客户未接受到任何信息，可能会使用户失去耐心。

4．Expires 属性

当一个 Web 页被传送到客户端浏览器后，IE 会将其内容保存到客户端计算机上，其目的是当以后遇到访问相同的 Web 页时，就直接从浏览器的缓冲区读取，这样可提高访问效率。Expires 属性指定在浏览器上缓冲存储的页面距离过期还有多少时间，以分钟为单位。如果用户在某个页过期之前又回到此页，就会显示缓冲区中的页面。

由于 ASP 应用的数据内容是动态变化的，所以使用过期页面得到的数据可能不真实。因此，若经常设置 response.expires=0，则可使缓存的页面立即过期。

【注意】该属性的设置必须放在 HTML 标记前，否则会出错。如果该属性在页中被多次设置，则使用最短的时间值。

5．ExpiresAbsolute 属性

ExpiresAbsolute 属性指定页面在浏览器缓存中的确切到期日期和时间。在未到期之前，若用户返回到该页，则应从缓存中提取页面显示。其语法格式如下：

Response.ExpiresAbsolute [= [date] [time]]

其中，date 参数指定页面的到期日期；time 参数指定页的到期时间。如果未指定时间，则该页面将在当天午夜到期。如果未指定日期，则该页面在脚本运行当天的指定时间到期。

【注意】若 ExpiresAbsolute 属性在页中被多次设置，则以最早到期的日期和时间为准。

4.2.2 Response 对象的方法

1．Write 方法

Write 方法是 Response 对象最常用的方法，用于向浏览器输出 HTML 信息。其语法格式如下：

Response.Write Variant

其中，Variant 可以是 VBScript 支持的任何类型的数据，通常为一个字符串，或是用连接符“&”连接在一起的字符串。

【例 4-3】 Write 方法的使用

```
------------------------------------------------------ex4-3.asp------------------------------------------------------
<%
Response.Write "<center><table width='100%'><td>"
Response.Write "欢迎你访问华东交通大学的教学网站<br>"
Response.Write "网站地址是<a href='http://cai.ecjtu.jx.cn/'>"&""""
Response.Write "cai.ecjtu.jx.cn"&""""&"</a><br>"
Response.Write "如果贵单位需要该教学系统，请与丁振凡老师联系<br>"
Response.Write "email 地址：zfding@ecjtu.jx.cn</td><table>"
%>
```

运行结果如图 4-4 所示。

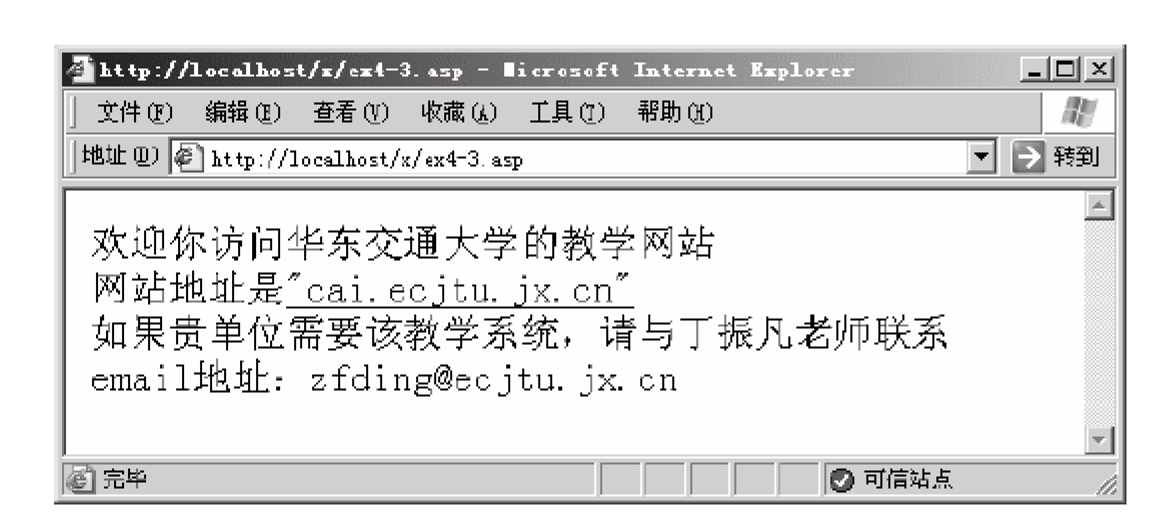


图 4-4　Response 对象的 Write 方法的使用

【说明】由于字符串用双引号作为分界符，所以在字符串中如果要给 HTML 标记的属性加引号应采用单引号。由于“%>”和“"”两个符号具有特殊的含义，因此输出的数据中不能直接包含字符“%>”和“"”，如果确实需要输出这两个符号，可用转义序列“%\>”来表示字符“%>”，使用两个相连的双引号字符表示输出一个双引号。

2．End 方法

End 方法使 Web 服务器停止处理脚本。如果 Response.Buffer 已设置为 True，则调用 Response.End 会立即将缓冲区中的内容发送给浏览器。

3．Flush 方法

如果 Response.Buffer 属性设置为 True，则执行 Flush 方法会立即将缓冲区中的内容发送给浏览器。如果 Response.Buffer 属性设置为 False，则该方法将导致运行时错误。

4．Redirect 方法

Redirect 方法使浏览器重定向到指定的 URL 网址。其语法格式如下：

Response.Redirect URL

【注意】使用 Response.Redirect 方法之前不能有任何输出内容发送给浏览器，因为它将引导浏览器打开新的网页。换句话说，该方法要放在程序中任何会产生输出的语句之前。

5．Clear 方法

Clear 方法用于删除缓冲区中存在的所有 HTML 内容。但要注意的是，如果未将 Response.Buffer 设置为 True，则该方法将导致运行时出现错误。如果要取消所有向客户端的输出，可首先清空缓冲区，然后调用 End 方法。以下代码在程序内出错时不会将错误报告给客户端。

```
<%
Response.Buffer=true
On error resume next      '设定遇到错误继续执行下一条语句
Err.clear %>
---------其他 HTML 和 ASP 代码 ----------
<%
if Err.number<>0 then
       Response.Clear
       Response.End
End if
%>
```

【说明】这里用到了 ASP 的错误处理机制，其相关内容在第 5 章中还将进一步介绍。

6．BinaryWrite 方法

BinaryWrite 方法用于给客户端发送非字符类型的数据。其语法格式如下：

Response.BinaryWrite 二进制数据

通常，BinaryWrite 方法要结合某种需要接受特殊非字符信息的定制 HTTP 客户应用程序一起使用。

4.2.3　Response 对象的数据集合

Response 对象只有一个 Cookies 数据集合，该数据集合允许将数据保存在客户端的硬盘中，如果指定的 Cookie 不存在，则创建新的 Cookie；相反，如果指定的 Cookie 存在，则更改 Cookie 的属性及值。其语法格式如下：

Response.Cookies("Cookiesname")[(Key)|. Attribute]= Value

其中：

- ❑ Cookiesname 为 Cookie 的名称。
- ❑ Key 是可选参数，如果指定了 Key，则表明 Cookie 是一个集合，将 Value 赋值给其中为 Key 的子键。
- ❑ Attribute 代表 Cookie 自身的某个属性，如表 4-4 所示。

表 4-4　Cookie 的属性

名　称	说　明
Domain	只写。若被指定，则只有某个网域可以存取该 Cookie
Expires	只写。指定 Cookie 的过期日期。为了在会话结束后将 Cookie 存储在客户端磁盘上，必须设置该日期。若此项属性的设置未超过当前日期，则在任务结束后 Cookie 将到期
HasKeys	只读。指定 Cookie 是否包含子键
Path	只写。若被指定，则 Cookie 将只发送到对该路径的请求中。如果未设置该属性，则使用应用程序的路径
Secure	只写。指定 Cookie 是否安全传送，即在传输过程中是否要加密

以下代码将显示客户请求中传递过来的所有 Cookie 变量的内容。

```
<%
For each cookie in Request.Cookies
if   Not cookie.HasKeys   then
      Response.Write cookie & "=" & Request.Cookies(cookie)
Else
   for each key in Request.Cookies(cookie)
      Response.Write cookie&"("&key&")"&"="& Request.Cookies(cookie)(key)
   next
end if
next
%>
```

Cookie 按其生命周期可划分为两种形式，即临时性 Cookie 和永久性 Cookie。临时性 Cookie 在浏览器关闭时自动过期，而永久性 Cookie 的过期时间取决于其 Expires 的属性设置。如果未设置 Expires 属性，则为临时性 Cookie。

为了方便 Cookie 的管理，可以将相关的 Cookie 用同一变量名表示，通过为 Cookie 设置子键来区分各个“子 Cookie”。注意，带有子键的 Cookie 的数据内容是一个集合。

【应用举例】在网络考试系统中分若干环节，包括组卷、试卷显示、试卷评分等。前面阶段组好的试题可以利用Cookie记录下来。一种办法是将选好的试题序号拼接为字符串，各题号间用逗号分隔。例如，以下将选取的单选题编号存入名字为 exam 且子键为 danxuan 的 Cookie 变量中。

```
Response.Cookies("exam")("danxuan") = "10,101,103,302,… "
```

在以后阶段，可以读取该 Cookie 变量的值，并利用 Split 函数将试题序号提取并存入数组中以便访问各道题。代码如下：

```
shitiString = Request.Cookies("exam")("danxuan")
shitiArray = Split(shitiString,",")
```

考试结束后要将相应的 Cookie 立即进行过期处理，避免在后续网页 HTTP 请求继续要传递该 Cookie 变量而带来不必要的开销。可以用以下方法让某 Cookie 变量立即过期。

```
Response.Cookies("exam").expires=now()
```

【例 4-4】　利用 Cookie 记录用户对本网页的访问次数和最近访问时间

```
----------------------------------------------------------ex4-4.asp----------------------------------------------------------
<%
if request.Cookies("visit")("num")="" then
    Response.Cookies("visit")("num")=1
else
    Response.Cookies("visit")("num")=request.Cookies("visit")("num")+1
end if
Response.Cookies("visit")("lastvisit")=now
Response.Cookies("visit").expires=date()+30
Response.Write "访问次数为："&request.cookies("visit")("num")&"<br>"
Response.Write "最近访问时间" &request.Cookies("visit")("lastvisit")
%>
```

在该例中，首先读取 Cookie 变量 Visit_num，检查用户端计算机是否保存有该名称的 Cookie 变量。如果有该变量，则说明用户已经访问过该页面，则将访问次数增加 1，同时输出访问次数。如果用户是首次访问该页面，则将 Cookie 变量 Visit_num 的值设置为 1 并写入客户计算机。

【应用实例】页面集合的共享处理。

某组页面集合需要在应用中多处用到，如网络教学测试子系统，在单元过关测试和网上考试中均要用到，只是抽取的试题不同。可以采用如下方法实现页面集的共享，让该

页面集执行完后，根据不同调用处的个性化要求返回到各自页面。

为了实现该目标，可以借助 Cookie 记录要返回点的 URL 地址。在进入页面集前设置要返回的 URL 地址。例如：

```
Response.Cookies("next_page")="student_main.asp"
```

而在页面集出口点的页面安排输出如下代码，让其自动执行客户端脚本返回目标页面。

```
Response.Write "window.location='"&request.cookies("next_page")&"';"
```

其中，window.location 代表浏览器窗体对象的地址设置。

4.3　Session 对象

前面已知道，可以利用 Cookie 保存客户的状态信息。但 Cookie 保存在客户端，与网站相关的所有 Cookie 的信息会随每次 HTTP 请求传递给服务器，不管当前处理程序是否需要这些 Cookie。显然，过量使用 Cookie 会占用大量的信息传递开销。而 Session 对象的引入可避免此类问题，Session 是将用户状态信息保存在服务器端。

客户如何与服务器端的 Session 取得关联呢？它是通过向客户端发送能代表 Session 对象唯一标识的一个 Cookie 对象来实现与 Session 关联的。当用户第一次请求 ASP 应用的某个页面时，服务器要检查 HTTP 头信息，查看是否有名为 ASPSESSIONID 的 Cookie 发送过来，如果有，则使用原来会话；否则服务器会启动一个新会话，并为该会话生成一个全局唯一的 SessionID，通过 Cookie 发送给客户端。每个用户依据各自的 SessionID，就可以访问存储在服务器上的属于自己的状态信息。

需要注意的是，会话状态仅在支持 Cookie 的浏览器中保留，如果客户关闭了 Cookie 选项，Session 也就不能发挥作用了。

4.3.1　Session 对象的属性

1．SessionID

SessionID 属性返回用户的会话标识。会话标识为长整数类型。

2．Timeout

Timeout 属性以分钟为单位为 Session 对象指定超时时限。如果用户在超时时限内不刷新或未发出新的网页访问请求，则其会话将终止。会话的超时时间默认为 20 分钟。可以通过 Internet 服务管理器中的相关设置改变默认超时限制值。在 ASP 程序代码中也可修改 Timeout 属性值。应依据 Web 应用程序的要求和服务器的内存空间来设置此值。过长的会话超时值将会导致打开的会话过多而耗尽服务器的内存资源。

【应用提示】考虑到 Timeout 的最大值限制，网络考试系统中通常用 Cookie 来保存用

户的状态信息，因为用户在解答试卷的过程中不会与服务器交互，考试时间可能较长。

4.3.2　Session 对象的方法

Session 对象仅有一个 Abandon 方法，用于删除所有存储在 Session 对象中的对象并释放这些对象的资源。当会话超时时，服务器也将删除 Session 对象。

在 Session 对象中既可以存储简单的数据类型的数据，也可以存储普通的对象引用。例如：

```
<%
Set rs=Server.createObject("ADODB.RecordSet")
Set Session("myrs")=rs
%>
```

【注意】在 ASP 中进行对象引用赋值时，前面要加上 Set 关键词。

另外，还可用 Session 对象来保存数组信息，Session 变量存储的是整个数组。例如：

```
<%
dim arr(4)
arr(0)="中国"
arr(1)="美国"
Session("myarray")=arr
%>
```

在后续页面中可将 Session 变量赋值给某个变量，从而通过该变量访问数组元素。例如：

```
<%
a = Session("myarray")
for each k in a
    Response.Write k & "<br>"
next
%>
```

4.3.3　Session 对象的事件

Session 对象的事件有 Session_OnStart 和 Session_OnEnd 两个，分别用于 Session 对象的启动和释放。

1．Session_OnStart 事件

Session_OnStart 事件在服务器创建新会话时发生，并且服务器在执行请求的页之前先处理事件的脚本代码。该事件是对 Session 变量进行初始设置的最佳时机。

为了确保用户启动会话时始终要打开某个“开始”页，可以在 Session_OnStart 事件脚本代码中检查用户打开的页是否是“开始”页，如果不是，就调用 Response.Redirect 方法转向“开始”页。程序代码如下：

```
<SCRIPT RUNAT=Server Language=VBScript>
Sub Session_OnStart
    startPage = "login.asp"
    currentPage = Request.ServerVariables("SCRIPT_NAME")
    if currentPage<>startPage then
        Response.Redirect(startPage)
    end if
End Sub
</SCRIPT>
```

【注意】在内置对象中，只有 Application、Server 和 Session 对象可以出现在 Session 事件的代码中。

2．Session_OnEnd 事件

Session_OnEnd 事件在会话被放弃或会话超时时发生。如果用户在指定时间内没有请求或刷新应用程序中的任何页，则会话将自动结束。在该事件代码中可安排会话结束时需要做的工作，通常安排清理系统对象、释放系统资源的脚本。

【注意】以上两个事件的代码必须包含在 Global.asa 文件中。

4.4　Application 对象

Application 对象是 Web 应用程序级的对象，这里的应用程序是由虚拟目录及其子目录下的所有.asp 文件构成的。每个 Web 站点可设置多个虚拟目录，也就是每个 Web 站点可以有多个 Web 应用。

Application 对象存储的数据可以被一个应用的所有用户共享，并可以在服务器运行期间持久地保存数据。以下代码显示了用 Application 对象存储字符串和对象的方法。

```
<%
Application("MyVar") = "Hello"
Set Application("MyObj") = Server.CreateObject("MyComponent")
%>
```

【注意】

（1）不能在 Application 对象中存储 ASP 内建对象。

（2）可将一个数组存储在 Application 对象中，但不能直接用 Application 访问数组元素。例如，下列脚本无法运行：

```
Application("StoredArray")(3) = "new value"
```

可先将 Application 存储的数组赋给某变量，然后通过该变量访问数组元素。例如：

```
Myarray = Application("StoredArray")
Myarray(3)= "new value"        '更改数组元素
Application("StoredArray")= Myarray
```

Application 对象与 Session 对象在数据存储与访问形式上相同，但本质上存在以下两点明显差异：

（1）作用范围不同。Session 对象是每个用户各自的，各用户同名的 Session 变量互不干涉；而 Application 对象是所有用户共享的，可以理解为应用级的全局变量。

（2）存活周期不同。Session 对象依赖用户的会话存在，一旦会话终止，则该用户的 Session 对象就被删除；而 Application 对象的存活周期跨度为整个应用的开始和停止时间。

4.4.1　Application 对象的方法

Application 对象为多用户共享，为了防止多个用户同时修改 Application 对象，保证数据的一致性和完整性，在修改 Application 对象时需要进行加锁。Application 对象提供两个方法用于加锁和解锁处理。

1．Lock 方法

Lock 方法用于锁定 Application 对象，只有取得了锁的那个用户的 ASP 程序才允许修改 Application 对象的值。

2．Unlock 方法

与 Lock 方法相反，Unlock 方法用于解除锁定，以便允许其他客户修改 Application 对象的属性。在用 Lock 方法锁定对象之后，如果用户没有明确调用 Unlock 方法，则服务器将在 ASP 程序执行结束或脚本执行出现超时时自动解除对 Application 对象的锁定。

【例 4-5】　用 Application 对象记录页面访问次数

```
-----------------------------------------------------ex4-5.asp----------------------------------------------------
<%
Application.Lock
if Application("NumVisits")="" then
    Application("NumVisits")=1
else
    Application("NumVisits") = Application("NumVisits") + 1
end if
Application.Unlock
Response.Write "你是本页的第" & Application("NumVisits") & "位访客 ！"
%>
```

【应用】将以上脚本嵌入到某个 ASP 文件中，就可为该页面添加一个计数器。

4.4.2　Application 对象的事件

Application 对象的事件有 Application_OnStart 事件和 Application_OnEnd 事件。

1．Application_OnStart 事件

Application_OnStart 事件是在 Web 服务器启动后，第一个用户访问首次请求应用程序

时发生，且它在第一个用户的 Session_OnStart 事件前发生。Application_OnStart 事件主要用于应用的初始化设置。事件处理方法形式如下：

```
Sub Application_OnStart
   事件处理代码
End Sub
```

2．Application_OnEnd 事件

Application_OnEnd 事件在服务器关闭时发生，用于实现应用的清理工作。事件处理方法形式如下：

```
Sub Application_OnEnd
   事件处理代码
End Sub
```

以上两个事件的代码均安排在 Global.asa 文件中。

4.4.3　Global.asa 文件

Global.asa 文件是一个可选文件，在该文件中安排 Application 对象和 Session 对象的事件脚本。注意，该文件必须放在 Web 应用程序的根目录下（包括虚拟目录）。

【例 4-6】　统计网站访问次数和当前在线人数

文件 1：Global.asa 文件

```
-------------------------------------------------------Global.asa-------------------------------------------------------
<script language="vbscript" runat="server">
Sub Application_OnStart()
     Application("count")=0
     Application("online")=0
end Sub
Sub Session_OnStart()
   Application("count")=Application("count")+1
   Application("online")=Application("online")+1
end Sub
Sub Session_OnEnd()
Application("online")=Application("online")-1
end Sub
</script>
```

文件 2：计数显示的 ASP 程序

```
--------------------------------------------------------ex4-6.asp--------------------------------------------------------
<table width="624" height="47" border="1">
<tr><td width="159">访问次数：</td>
<td width="449"><%=Application("count")%></td></tr>
<tr><td>在线人数：</td><td><%=Application("online")%></td></tr>
</table>
```

【说明】由于 Application 对象的生命周期特点，这里统计的用户访问次数均以应用的启动和停止为界。如果要考虑历史访问的情形，则要用文件或数据库存储数据。可以在 Application 启动时读取历史数据，而在 Application 停止时将数据写入文件或数据库。

【例 4-7】　简单在线讨论区编程

以下为一个在线讨论区的简单实现。在用户进入讨论区前先输入一个“昵称”，在讨论区中显示在线用户个数，假定退出讨论区时要求用户单击“退出”超链接。讨论区显示的发言条目为最近的 15 条发言。帖子和当前在线用户信息分别用 message 和 user 两个 Application 变量存储。

为了让用户及时看到讨论区上的发言，必须让显示讨论区内容和用户列表的页面保持自动定时刷新。页面自动刷新处理办法之一是在网页的 head 部分加入如下 HTML 标记行：

```
<meta http-equiv="refresh" content="5">
```

此时即让网页每隔 5 秒更新一次。

程序 1：讨论区登录页面

讨论区登录页面获取用户标识，并将该标识保存在 Cookie 变量中。

```
-------------------------------------------------------login.asp-------------------------------------------------------
<%
if request("name")<>"" then
    user= request("name")
    Response.Cookies("username")= user
    application.lock
    '将用户名加入在线用户中
    application("user")= user +"<br>"+application("user")
    application.unlock
    Response.Redirect("enter.asp")
End if
%>
<h2 align="center">在线讨论区</h2>
<form name="f1" action="login.asp" method="post">
<div align="center">
请输入用户名：<input type="text" name="name" size="10">
<input type="submit" name="b1" value="登录"></div>
</form>
```

运行结果如图 4-5 所示。

【说明】本例表单提交和表单生成为同一 ASP 文件，也就是采用自提交方式，这样设计的优点是可减少 ASP 文件的数量，便于应用管理。但缺点是容易出现代码设计上的考虑不足。原因是在同一文件中既要考虑未提交前表单的生成，还要考虑表单提交的处理。程序中将登录名保存到名为 username 的 Cookie 变量中，并通过字符串拼接存储到 application ("user")变量中，然后转向讨论区主界面。



图 4-5　讨论区登录页面

程序 2：讨论区的框架页面

```
------------------------------------------------------------enter.asp------------------------------------------------------------
<% if request.cookies("username")= "" then
      Response.Redirect("login.asp")
      end if %>
<frameset rows="80%,*" frameborder=0 framespacing=0>
<frame name="up" src="message.asp" >
<frame name="down" src="speak.asp">
</frameset>
```

【说明】在该页面首先检查用户是否登录过，若未登录，则转向登录页面。该页面由上下两个框架构成，上面的一个框架显示讨论区内容和在线用户，下面的框架含有输入发言的表单。

程序 3：输入发言的处理页面

```
------------------------------------------------------------speak.asp------------------------------------------------------------
<%
if request("speak")="发言" then
      name = request.cookies("username")
      content = request.form("content")
      application.lock
      application("message")=name+":"+content+"<br>"+application("message")
      application.unlock
      show=application("message")
      '确保显示最近的 15 条信息，其他内容删除
      i=1
      for n=1 to 15
         i=instr(i,show,"<br>")+3
         if   i=3 then exit for
      next
      if i<>3 then
         application.lock
         application("message")=left(show,i)
         application.unlock
      end if
end if
%>
<form method="post" action="speak.asp">请发言：
```

```
<input type="text" name="content" size="30">
<input name="speak" type="submit" value=" 发 言 ">  
<a href="logout.asp" target="_parent">退出</a>
</form>
```

【说明】这是一个自提交处理页面，将输入的发言通过字符串拼接的办法写入名为 message 的 Application 变量中，并清理其中的内容，而只保存最近的 15 条发言。

程序 4：发言和在线用户显示页面

```
----------------------------------------------------message.asp----------------------------------------------------
<html>
<head>
<META HTTP-EQUIV="REFRESH" CONTENT="5">
</head>
<body>
<center><h1>在线讨论区</h1></center>
<table width="100%">
<td width="80%">
<%=application("message")%>
</td>
<td width="20%" >
<div align=center>在线用户：</div><hr>
<%=application("user")%>
</td>
</body>
</html>
```

程序 5：用户退出处理程序

```
------------------------------------------------------logout.asp------------------------------------------------------
<%
x= application("user")
user=request.cookies("username")
pos=instr(x,user)
x=left(x,pos-1)+mid(x,pos+len(user+"<br>"))
application.lock
application("user")=x
application.unlock
Response.Redirect("login.asp")
%>
```

【说明】用户退出要解决的问题是将用户从 Application 变量 user 中删除，同时离开讨论区，并重定向到登录页面。

程序 6：Global.asa 文件

```
------------------------------------------------------Global.asa------------------------------------------------------
<script language=VBScript runat=server>
sub Application_OnStart
  application("user")=""
```

```
  application("message")=""
end sub
</script>
```

【说明】利用 Application_OnStart 事件解决 Application 对象的初值设置，此时也可以不使用该文件，因为这里设置的初值均为空串，与变量默认值是相同的。

运行结果如图 4-6 所示。



图 4-6　讨论区发帖界面

4.5　Server 对象

Server 对象提供了一系列方法以实现典型操作的使用，以及对服务器功能的扩展。Server 对象的语法格式如下：

Server.属性|方法

4.5.1　Server 对象的属性

Server 的属性只有 ScriptTimeout，该属性用于设置一个 ASP 程序所允许的最长执行时间，在脚本运行超过这一时间之后即作超时处理，其设置以“秒”为单位，系统的默认值为 90 秒。其语法格式如下：

Server.ScriptTimeout=秒数

ScriptTimeout 属性不能设置为小于系统指定的默认值，否则仍以默认值为准，当然也可以在 IIS 管理器中修改系统默认值。

4.5.2　Server 对象的方法

1．CreateObject 方法

CreateObject 方法用于创建已经注册到服务器上的 ActiveX 组件实例。这是一个非常重要的方法，使用该方法可以使 ASP 程序利用服务器的组件不断扩展新的功能，如数据库的

存取、文件存取、广告轮显、文件上传等。CreateObject 方法的语法格式如下：

Server.CreateObject("ActiveX 组件")

这里的 ActiveX 组件可以是 ASP 的所有内置组件，也可以是第三方组件，但不能是 ASP 内置对象。ASP 的常用内置组件有文件访问组件（Scripting.FileSystemObject）、广告轮显组件（MSWC.AdRotator）、内容链接组件（MSWC.NextLink）、计数器组件（MSWC.Counters）。后面 3 个组件功能相对有限，文件访问组件功能很强，它可以对服务器上的文件和目录进行管理和访问。黑客常利用文件访问组件破坏系统，为了减少系统漏洞，常将该组件设为禁用。

为了实现对数据库的访问，ASP 提供了 ADO 系列组件，例如，下面的代码用于创建数据库连接对象。

```
Set Conn = Server.CreateObject("ADODB.Connection")
```

如果创建的对象不再需要使用，可以用如下方法释放所占用的资源。

```
Set Conn = nothing
```

2．MapPath 方法

MapPath 方法将指定的虚拟路径（相对路径）映射到物理路径（即真实路径）。这在实际应用中是很有用的，一般情况下，文件路径用虚拟路径来表示，但有时必须使用物理路径，如文件上传或对服务器上的文件进行读写操作时必须使用物理路径。

MapPath 方法的语法格式如下：

Server.MapPath("虚拟路径")

如果 Path 以一个正斜杠（/）或反斜杠（\）开始，则 MapPath 方法以 Web 应用的根目录（含虚拟目录的根）作为相对点；如果 Path 不是以斜杠开始，则 MapPath 方法以当前 ASP 文件的路径作为相对点。

例如，test.asp 文件和数据库文件 data.mdb 都位于目录 C:\Inetpub\wwwroot\下，在 test.asp 文件中执行如下脚本：

```
<%= Server.MapPath("cai\data.mdb")%>
```

则相应的输出结果为 c:\inetpub\wwwroot\cai\data.mdb。

【注意】Server.MapPath 方法并不检查指定路径和文件是否存在，路径的正确性由程序员自己把握。

3．URLEncode 方法

在 URL 字符串中不允许出现空格等特殊字符，使用 URLEncode 方法可以使这些字符加入 URL 时转换成 URL 中等效的字符。例如，空格用“+”代替，ASCII 码大于 126 的字符用“%”后跟十六进制代码进行替换。在实际应用中，经常用 URL 参数为超链接的目标传递动态信息，可以采用 URLEncode 方法进行数据包装处理。例如：

```
<% Response.Write Server.URLEncode("x.asp?user=Nace Coer") %>
```

将得到的结果是 x%2Easp%3Fuser%3DNace+Coer。

4．HTMLEncode 方法

HTMLEncode 方法是对指定的字符串进行 HTML 编码。其语法格式如下：

Server.HTMLEncode(string)

如果想在目标页面中显示 HTML 文本或 ASP 脚本代码，需要用 HTMLEncode 方法。例如，以下代码能在页面上看到
的文字，而不是换行效果。

```
<% Response.Write server.HTMLEncode("<br>能看见吗?")%>
```

【应用实例】在很多数据录入应用中，如试题库、网上作业、讨论等，从文本框输入的数据需要按照原样显示，因此可以使用 HTMLEncode 方法将数据库中读取的内容进行变换后输出显示。

5．Transfer 方法

Transfer 方法的作用是将 ASP 程序的当前控制权转移到另一个指定的 ASP 程序。其功能类似于 Response 对象的 Redirect 方法。两者的主要差别如下：

（1）执行过程和目标对象不同。当执行 Response.Redirect 方法后，服务器将方法指定的地址发送给浏览器，由浏览器请求新地址，所以 Response.Redirect 方法的目标地址可以是任何 URL 地址；而执行 Server.Transfer 方法是直接由服务器将代码执行流程转向指定的 ASP 文件，目标地址只能是同一站点的当前目录或子目录下的页面文件。

（2）传递的 URL 参数的数据量不同。Response.Redirect 方法传递的数据量受浏览器地址栏的限制，最大为 2K，而 Server.Transfer 方法传递的数据量可超过 2K。

本 章 小 结

本章介绍了 ASP 的 Response、Request、Session、Application、Server 等内置对象的基本使用方法，读者要熟练掌握这些对象的属性和方法的使用。具体包括 Request 对象和 Response 对象的使用方法；利用 Request 对象的数据集合获取客户提交数据的方法；利用 Response 对象的给客户浏览器发送响应信息和重定向的方法；利用 Cookie 和 Session 保存用户的状态信息的方法；利用 Application 对象实现信息的共享访问的方法；了解 Global.asa 文件中 Session 和 Application 对象的事件编程；熟悉 Server 对象的常用方法的使用特点等。

习　　题

1．选择题

（1）若要停止 ASP 程序的执行并将存放在缓冲区的输出传送至浏览器端，可以使用 Response 对象的（　　）方法。

A．Clear　　B．Flush
C．End　　D．Write

（2）若要将浏览器端导向至其他网页，可以使用 Response 对象的（　　）方法。

A．Redirect　　B．End
C．Flush　　D．AppendToLog

（3）下列有关 Response.Write 方法的说法正确的是（　　）。

A．若要显示的信息包含双引号，必须将双引号“"”改为单引号“'”
B．若要显示的信息包含双引号，必须将双引号“"”改为两个双引号“""”
C．若要显示的信息包含%>，必须改为\%>
D．若要显示的信息包含%>，必须改为%\>

（4）使用 Response 对象的（　　）属性可设置放进缓存区的网页的逾期时间长短。

A．CacheControl　　B．Buffer
C．Status　　D．Expires

（5）关于 Cookie 的缺点，下列说法正确的是（　　）。

A．造成浏览器端有安全上的威胁
B．Cookie 会自动消失
C．Cookie 可以记录对象、数组等复杂的数据类型
D．Cookie 可能被禁止写入浏览器端

（6）下列（　　）环境变量可以返回服务器端的 IP 地址。

A．Path_Info　　B．Local_Addr
C．Remote_Addr　　D．URL

（7）Application 对象只能记录变量、字符串、日期等简单的数据类型，此说法（　　）。

A．正确　　B．不正确

（8）若要在完成目前的网页之后便结束 Session 对象，可以使用（　　）方法。

A．Clear　　B．End
C．Timeout　　D．Abandon

（9）下列语句中说法正确的是（　　）。

A．Session 对象无法记录数组、对象等复杂的数据类型
B．Session 对象存储在服务器的内存，一旦有很多浏览器进行联机，效率将会降低
C．无论在何种情况下，SessionID 都是唯一的
D．Timeout 属性的值越大，所占用的内存越多

（10）下列叙述错误的是（　　）。

A．Application 对象的 OnStart 事件发生于 Web 服务器开始执行时
B．Session 对象的 OnEnd 事件发生于浏览器与服务器断线时，或者浏览者在 Session.Timeout 指定的时间内没有访问网页时
C．若要设置 Session 对象的初始值，可以在 Global.asa 文件内进行设置

D．Global.asa 文件应放在和网页相同的文件夹内

（11）若要将字符串进行编码，使它不会被浏览器解释为 HTML 语法，可以使用 Server 对象的（　　）方法。

A．HTMLEncode　　B．URLEncode

C．MapEncode　　D．ASPEncode

2．编程题

（1）编写 ASP 程序查看今天的日期，根据奇偶性不同显示不同风格的页面。

（2）编写 ASP 程序显示系统目前的时间，包括日期、时、分、秒。

（3）编写 ASP 程序，限制用户 IP 地址为某个地址区间的允许访问网页（如以 192.168.0 开头的），显示欢迎信息，否则显示“禁止访问”。

（4）编写用户登录页面，只有程序中指定的用户名、密码才能登录成功。在登录成功后转向的页面中显示“×××用户，欢迎你使用本系统”。分别采用 Cookie 变量或 Session 变量保存用户名给后续页面访问。

（5）编写程序实现如图 4-7 所示的个人小档案调查的表单界面，并编写表单处理程序将表单提交后用户填写的数据显示出来。



图 4-7　个人小档案调查的表单界面

（6）编写一个网上职业投票应用，选出大众最喜欢的职业。利用 Application 对象记录每个行业的得票数量。此时可以假设几个职业作为投票调查对象。

第 5 章　ASP 访问数据库

在实际应用系统中，通常用数据库存储数据信息。本章将介绍如何在 ASP 脚本代码中通过 ADO 对象访问关系数据库表格中的数据。作为本章的基础，首先对关系数据库中的查询语言——SQL 语言进行简要介绍。

5.1　结构化查询语言 SQL

SQL 语言是一种通用的数据库查询语言，目前绝大多数的数据库均支持这种查询语言，如 Access、MySQL、SQL Server、Oracle、SyBase 等。SQL 语言已成为用户与基于 SQL 的 DBMS 的接口，其主要功能包括以下方面。

（1）数据定义功能：用于定义数据库表格中数据的组织和结构。

（2）数据检索功能：用于从数据库中检索数据。

（3）数据操纵功能：可以更改数据库内容，包括增加、删除和修改等操作。

5.1.1　SQL 命令的基本构成

SQL 命令主要由命令、子句、运算符和函数等构成。命令可以分为数据定义语言（DDL）和数据操纵语言（DML）两大类。数据定义语言用于创建或修改数据库、表索引文件，数据操纵语言用于对数据进行查询、修改、删除和插入操作。需要注意的是，SQL 命令不区分大小写，在命令中使用的字符串常量用单引号或双引号括起来，日期常量用单引号括起来。

1．SQL 命令及子句

（1）数据定义命令

- CREATE：用于创建数据库、表和索引。
- DROP：用于删除数据库及数据库中的表和索引。
- ALTER：用于修改表的结构。

（2）数据操纵命令

- SELECT：在数据库表格中查找满足条件的记录。
- INSERT：在数据库表格中插入新的记录。
- UPDATE：修改数据库表格中的记录。
- DELETE：删除数据库表格中的记录。

（3）命令子句

命令子句用来指定查询条件、数据的来源、数据的组织排列方式等，常用的命令子句如下。

- FROM：指定数据来源，一般为表格列表。
- WHERE：指定选择记录时要满足的条件。
- GROUP BY：设置所选择的记录进行分组的方式。
- ORDER BY：用于指定记录按哪个字段进行排序，以及排序方式是升序还是降序。

2．在 SQL 命令中使用的运算符与函数

在 SQL 命令中主要用 3 种运算符，即算术运算符、逻辑运算符和关系运算符。

（1）算术运算符

该运算符包括加（+）、减（-）、乘（*）、除（/）和取余运算（%）。

（2）逻辑运算符

该运算符包括 AND、OR 和 NOT 3 种，分别代表逻辑与运算、逻辑或运算和逻辑取反运算。

（3）关系运算符

除了一般程序设计语言中常见的关系运算符，如>（大于）、<（小于）、>=（大于等于）、<=（小于等于）、<>或!=（不等于）、=（等于）外，还有如下几个特殊的关系运算符。

- BETWEEN…AND：用于指定字段值的范围。
- IN：指定字段的可能取值。
- LIKE：在模式匹配中使用，实现模糊查找。

SQL 提供了丰富的函数，在汇总查询部分将介绍几个常用函数，读者若想了解更多的函数，可参见相关书籍。

5.1.2　SQL 查询

查询是 SQL 语言的核心，用于表达 SQL 查询的 SELECT 语句是功能最强也是最复杂的 SQL 语句。该语句将从指定的表中检索出满足条件的记录，被检索出来的记录形成一个记录集合，简称为记录集。

1．简单 SQL 查询

SELECT 查询语句由 7 个子句构成，其中 SELECT 和 FROM 子句是必须要有的，其他子句可以根据需要任意选择。其语法格式如下：

```
SELECT   [ALL|DISTINCT]   <关系表字段(表达式)列表|*>
FROM   <关系表名(别名)列表>
[WHERE  查询条件]
[GROUP BY  分组要求]
[HAVING    分组搜索条件]
[ORDER BY      排序要求]
[INTO <新关系表名>]
```

【说明】

- SELECT 子句：列出查询的数据项。这些项可能取自数据库中表的列，也可以是 SQL 在执行查询时需要进行计算的表达式，特别是使用“*”时则表示包含指定表中所有的列。其中，ALL 和 DISTINCT 选项表示查询结果中是否允许有内容重复的行，默认是 ALL 选项，表示允许有内容重复的行，DISTINCT 选项将不含内容重复的行。
- FROM 子句：列出查询所涉及的数据表。
- WHERE 子句：设置对表的查询条件。
- ORDER BY 子句：确定查询结果按指定的一列或多列中的数据进行排列。默认是不进行排序的。
- GROUP BY 子句：指定当前查询是汇总查询，即不是根据每行产生一个查询结果，而是对相似的行进行分组，然后再对每组产生一个汇总查询结果。
- HAVING 子句：根据该子句中给出的条件表达式对 GROUP BY 所得到的分组结果进行过滤，从而选择出满足条件的组。
- INTO 子句：确定是否将查询出的结果存入一张新关系表中。

在网络教学系统中，设有一个 userTable 表，包含有用户标识、密码、姓名、班级名、性别等字段，其中用户标识为关键词字段，具有唯一性。以下为使用 SELECT 查询语句的若干示例。

下面的 SQL 语句用于查询 userTable 表中所有用户的姓名、性别。

```
SELECT 姓名,性别 FROM  userTable
```

下面的 SQL 语句可列出 userTable 表中所包含的班级名称。

```
SELECT DISTINCT 班级名 FROM userTable
```

【说明】相同的班级只出现一次。

2．特殊查询处理

SQL 提供了丰富的查询搜索条件，使用户能够方便地选取数据。

（1）模糊匹配查找

模糊匹配查找用于检验一个包含字符串数据的列的值是否匹配一指定的模式。通常使用“%”和“_”两个通配符，“%”可以和零个或任意长度的字符串相匹配；“_”可以和任何单字符串相匹配。模糊匹配查找的语法格式如下：

[NOT] LIKE 模式

下面的 SQL 语句用于查询“单选题”表格的 content 字段中含有 Java 的所有试题。

```
SELECT  * FROM 单选题 WHERE content LIKE  "%Java%"
```

（2）多表查询处理

SELECT 允许将多张关系表的数据联系在一起进行搜索查询。在 SQL 中进行多表查询时，查询表中的关键字通过关联与另一个表中代表相同信息的字段建立联系，从而将要查

询的多张表逻辑上连接成为一张大表，再对所产生的大表进行与单表查询类似的处理。多表查询时，如果某个字段在多个表中出现过，则要在字段名通过句点分隔加上所属表的约束，如 userTable.username 表示 username 字段来自 userTable 表。

例如，网上作业子系统用来实现网络作业的布置与学生解答，需要进行多表设计，包括作业布置表（task）和作业解答表（taskanswer）。作业布置表包括作业编号、标题、内容、涉及班级等字段，作业解答表包括作业编号、学生标识、解答等字段。这两个表通过“作业编号”字段建立联系。

假设要查询“学生标识”为 jsj0810 的学生对“作业编号”为 101 的作业的解答信息，则可用如下 SQL 语句表示：

```
SELECT 标题, 内容, 解答 From task, taskanswer
where task.作业编号= taskanswer.作业编号
and task.作业编号=101 and taskanswer.学生标识='jsj0810'
```

（3）组属测试运算符

组属测试运算符用于测试一个表达式的值是否属于某一组值。其语法格式如下：

[NOT]　IN(数据列表)

下面的 SQL 语句查询 userTable 表中属于“计算机 09-1”和“计算机 09-2”班的用户。

```
SELECT  *  FROM  userTable
WHERE  班级名 IN ('计算机 09-1','计算机 09-2')
```

此时，IN 后面的数据列表也可以是另一个 SELECT 语句的查询结果产生的列表。例如，列出所有非历史班级的学生。不妨假设另有一个“班级表”存储所有班级的班级名，其中还包括一个名为“历史”的布尔型字段表示是否为历史班级。

```
SELECT  *  FROM  userTable
WHERE  班级名 IN (select 班级 from 班级表 where 历史=false)
```

【说明】“历史=false”写成“NOT 历史”更常见。

（4）查询结果排序

使用 ORDER BY 子句可对查询结果进行排序。其语法格式如下：

ORDER BY <字段名或列序号　ASC/DESC>,…<字段名或列序号　ASC/DESC>

其中，ASC 代表升序，DESC 代表降序。

下面的 SQL 语句将列出“教材”表中的全部教材，并按数量降序排列，若数量相同，则再按单价升序的顺序排列。

```
SELECT * FROM 教材 ORDER BY 数量 DESC, 单价 ASC
```

（5）使用范围测试运算符

BETWEEN…AND 操作符用于测试一个表达式的值是否在某一个范围之内。其语法格式如下：

[NOT] BETWEEN 值较小的表达式 AND 值较大的表达式

下面的 SQL 语句将列出“教材”表中单价在 30～50 元之间的教材名称、单价、作者、

出版社。

```
SELECT 教材名称, 单价,作者,出版社 FROM 教材
WHERE 单价 BETWEEN 30 AND 50
```

【说明】范围测试运算符使用不是很广，它可以用关系运算符替代。例如 A BETWEEN B AND C 的格式完全等价于(A>=B)AND(A<=C)，只是用 BETWEEN…AND 显得更直观些。

（6）汇总查询

汇总查询用于对满足查询条件的数据进行多种计算处理，并将处理结果作为查询结果输出。SQL 提供了 6 种汇总处理函数。

- ❑ SUM()：用于计算表中某一列的总和。
- ❑ AVG()：用于计算表中某一列的平均值。
- ❑ MIN()：用于选择表中某一列的最小值。
- ❑ MAX()：用于选择表中某一列的最大值。
- ❑ COUNT()：用于计算表中某一列中值的个数。
- ❑ COUNT(*)：用于计算某张表的行数。

由于上述 6 个函数均是对表中的某一列进行计算处理，因此它们也称为列函数，这些函数的参数均是表中的列名。

下面的 SQL 语句查询“教材”表中全部教材的数量及平均单价。

```
SELECT SUM(数量)  AS '总数量' , AVG(单价)  AS  '平均单价'  FROM 教材
```

其中，AS 子句用于为汇总统计结果的列命名。

下面的 SQL 语句查询 userTable 表中的用户总数。

```
SELECT  COUNT(*)  AS  '用户总数'  FROM  userTable
```

【说明】在进行数据表中行的计数时，通常用 COUNT(*)。

（7）分组查询处理

分组查询对查出的数据按某些指定字段的值进行分类，并将查询结果按行组的形式输出。同一行组中，所指定字段的值相同，而不同行组中相应字段的值不同。SELECT 语句中的 GROUP BY 子句指定分组查询用到的字段。

下面的 SQL 语句查询 userTable 表中每班的人数。

```
SELECT 班级名,COUNT(班级名)  AS  '人数'
FROM userTable GROUP BY 班级名
```

【说明】如果 SQL 的列函数和 GROUP BY 子句连用，则表示 SQL 将查询出的结果分组，再用列函数分别作用于每个行组，并对每个组产生一个计算结果。

分组查询中有一个 HAVING 子句用于指定对分组进行搜索的条件，以查出满足一定要求的行组。HAVING 子句的语法格式与 WHERE 子句类似，但发生作用的时间点不同，WHERE 子句是在分组之前检查有哪些记录有资格参加分组统计，而 HAVING 子句是对分组统计后的结果再次进行筛选，从而形成最后的结果记录集。

下面的 SQL 语句查询 userTable 表中人数少于 20 的班级。

```
SELECT 班级名,COUNT(班级名)  AS  '人数'  FROM userTable
GROUP BY 班级名
HAVING  COUNT(班级名)<20
```

5.1.3　其他 SQL 语句

1．表结构维护

（1）表的创建

CREATE TABLE 语句用于创建新表格，其语法格式如下：

CREATE TABLE 表名称(字段名称 1type, 字段名称 2　type,…)

其中，type 规定了列的数据类型。不同数据库的数据类型表示有一定的差异，表 5-1 列出了 SQL Server 数据库中常用数据类型的存储及取值特点。

表 5-1　SQL Server 数据库中常用数据类型的存储及取值特点

数 据 类 型	存储占字节数	描　　述
tinyint	1 字节	整数，取值范围为 0～255
smallint	2 字节	整数，取值范围为-32768～32767
int	4 字节	整数，取值范围为 -2^{31}～$2^{31}-1$
bigint	8 字节	整数，取值范围为 -2^{63}～$2^{63}-1$
decimal (p,s) numeric (p,s)	5～17 字节	固定精度和小数位数的数值数据类型。p 参数规定最多可存储的十进制数字的总位数，s 参数规定小数点后的最大位数
real	4 字节	实数，取值范围为-3.40E+38～3.40E+38
float(n)	4 或 8 字节	存储 1～53 的可变精度的浮点值，参数 n 代表精度值。n 为 1～24 时，占用 4 字节存储空间；n 为 25～53 时，占用 8 字节存储空间
char(size)	占 size 个字节	固定长度的字符串（可容纳字母、数字以及特殊字符）。size 规定字符串的长度
varchar(size)	取决于实际字符数	可变长度的字符串（可容纳字母、数字以及特殊的字符）。size 规定字符串的最大长度
date	3 字节	存储日期

例如，以下 SQL 语句创建一个名为 Users 的表，包含 userid、userName、address、age 4 个字段。

```
CREATE TABLE Users (userid int, userName varchar(255), address varchar(255), age int)
```

（2）表的删除

格式：drop table 表名称

【注意】删除一个表时也会将表的结构及索引删除。

（3）表结构修改

❑　增加字段

命令格式：Alter table 表名称 add column 字段名称 栏类型

❑ 添加主键

命令格式：Alter table 表名称 add primary key(字段名称)

【说明】将指定的字段定义为主键。

要删除某主键，可使用如下命令：

```
Alter table 表名称 drop primary key(字段名称)
```

（4）索引的创建与删除

❑ 创建索引

命令格式：create [unique] index 索引名 on 表名称(字段列表)

【说明】按指定的字段列表中的字段顺序建立索引，unique 要求索引字段的值在数据表中不能重复出现。

❑ 删除索引

格式：drop index 索引名 on 表名称

【注意】索引是不可更改的，若想对其进行更改，则必须删除并重新创建。

2．数据更新

SQL 是一种完备的数据库语言，它不仅可用于数据库的查询，还可用于数据库中数据的修改和更新。与 SELECT 查询语句相比，更改数据库内容的 SQL 语句显得较为简单。在 SQL 中有 3 条用于数据更新的语句，即 INSERT、DELETE 和 UPDATE 语句，分别用于数据库数据的添加、数据的删除和数据的更改操作。

（1）数据的添加

INSERT 语句用于向一关系表中添加一行新数据。其语法格式如下：

INSERT INTO 表名(字段名列表)VALUES(字段值列表)

【说明】要求字段值列表中的各数值顺序及数据类型与字段名列表中的各字段名相互对应，否则将会出现操作错误的提示。

例如，以下 SQL 语句向 userTable 表中插入一个新记录。

```
INSERT INTO userTable VALUES（'jsj0911','5423','张三','计算机 09-1','男'）
```

在向表中插入记录时，可以只列出部分字段名，例如，以下命令向 userTable 表中插入一个新记录。

```
INSERT INTO userTable(用户标识,密码,姓名)VALUES('guest','123','来宾')
```

【注意】只要是允许为空和有默认值的字段名都可以省略，但不允许为空的字段不能省略，否则将出现执行错误。

（2）数据的删除

DELETE 语句用于删除关系表中一行或若干行的操作。其语法格式如下：

DELETE　FROM 表名 [WHERE<搜索条件>]

例如，以下 SQL 语句从 userTable 表中删除用户标识以 123 开头的记录。

```
DELETE FROM userTable WHERE 用户标识 like '123%'
```

（3）数据的修改

UPDATE 语句用于修改关系表的选定行中一列或若干列的值。其语法格式如下：

UPDATE　表名　SET　字段名=表达式,…,　字段名=表达式　[WHERE<搜索条件>]

以下 SQL 语句将 userTable 表中“用户标识”为 jsj081 的密码改为 123。

```
UPDATE userTable   SET  密码='123'   WHERE   用户标识='jsj081'
```

以下 SQL 语句将“教材”表中清华大学出版社的库存数量超过 500 本的图书单价降低 30%。

```
UPDATE  教材   SET Price=Price-Price*0.3
WHERE  出版社= "清华大学出版社" AND   数量>500
```

5.2　ADO 对象模型简介

5.2.1　ADO 内幕

ADO（Activex Data Object，ActiveX 数据对象）是微软提供的新一代数据库存取访问技术，其是在 OLE DB 技术的基础上实现的。利用 ADO 对象，通过 ODBC 驱动程序或 OLE DB 链接字符串，可实现对任意数据库的。存取和访问。ODBC 是 Open DataBase Connectivity 的缩写，称为开放式数据互联。有了 ODBC 驱动程序，就可以用同样的方法访问任何符合 ODBC 标准的关系型数据库。ADO、OLE DB、ODBC 以及各种数据库的关系如图 5-1 所示。

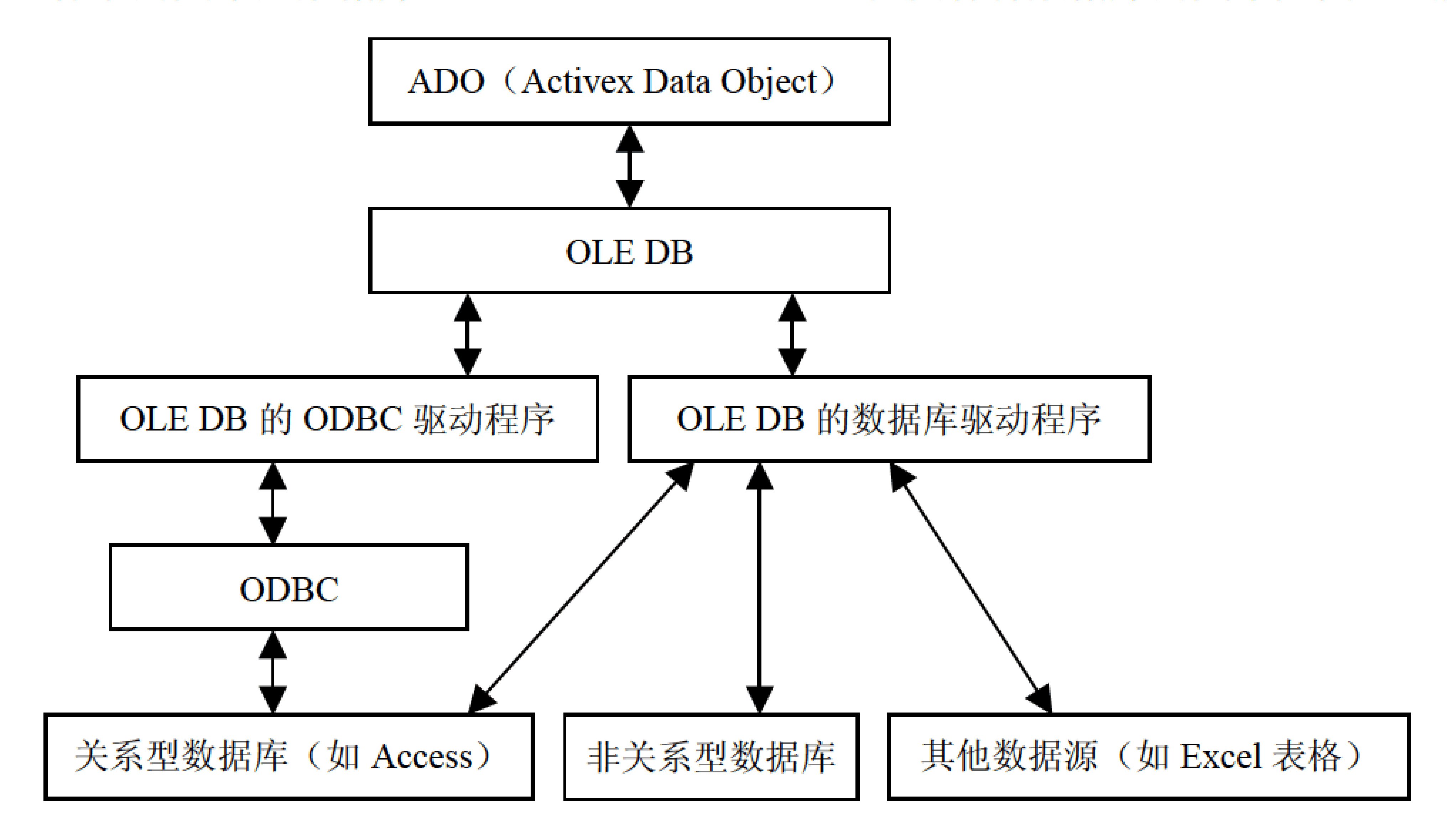


图 5-1　ADO、OLE DB、ODBC 以及各种数据库的关系

通过图 5-1 可以看出，ADO 可以通过 ODBC 或 OLE DB 两种驱动程序与数据库进行通信。从执行效率上看，采用 ODBC 的环节明显要多于 OLE DB，因此，直接用 OLE DB 驱

动程序访问数据库效率更高。

5.2.2　ADO 对象和数据集合

ADO 技术是通过 ADO 对象的属性和方法来完成相应数据库访问的。ADO 共有以下 7 个独立对象。

（1）Connection 对象：用于创建 ASP 程序与数据源之间的连接。

（2）Command 对象：用于定义对数据源执行的命令，包括 SQL 命令、存储过程等。

（3）Recordset 对象：用于封装查询的结果，称为结果集。

（4）Field 对象：用于表达一行结果中各子段的类型和值。

（5）Error 对象：用于检测和判断在数据库操作中出现的错误，如连接失败。

（6）Property 对象：用于描述 ADO 对象的动态特性。

（7）Parameter 对象：用于描述 Command 对象的参数。

ADO 包含 4 个数据集合：Fields 集合是与一个 Recordset 对象关联的所有 Field 对象的集合；Parameters 集合是与一个 Command 对象关联的所有 Parameter 对象的集合；Errors 集合包含一个连接上所产生错误的所有 Error 对象的集合；Properties 集合是所有 Property 对象的集合，该集合与 Connection、Command、Recordset 等对象关联。

ADO 对象与数据集合之间的关系如图 5-2 所示。

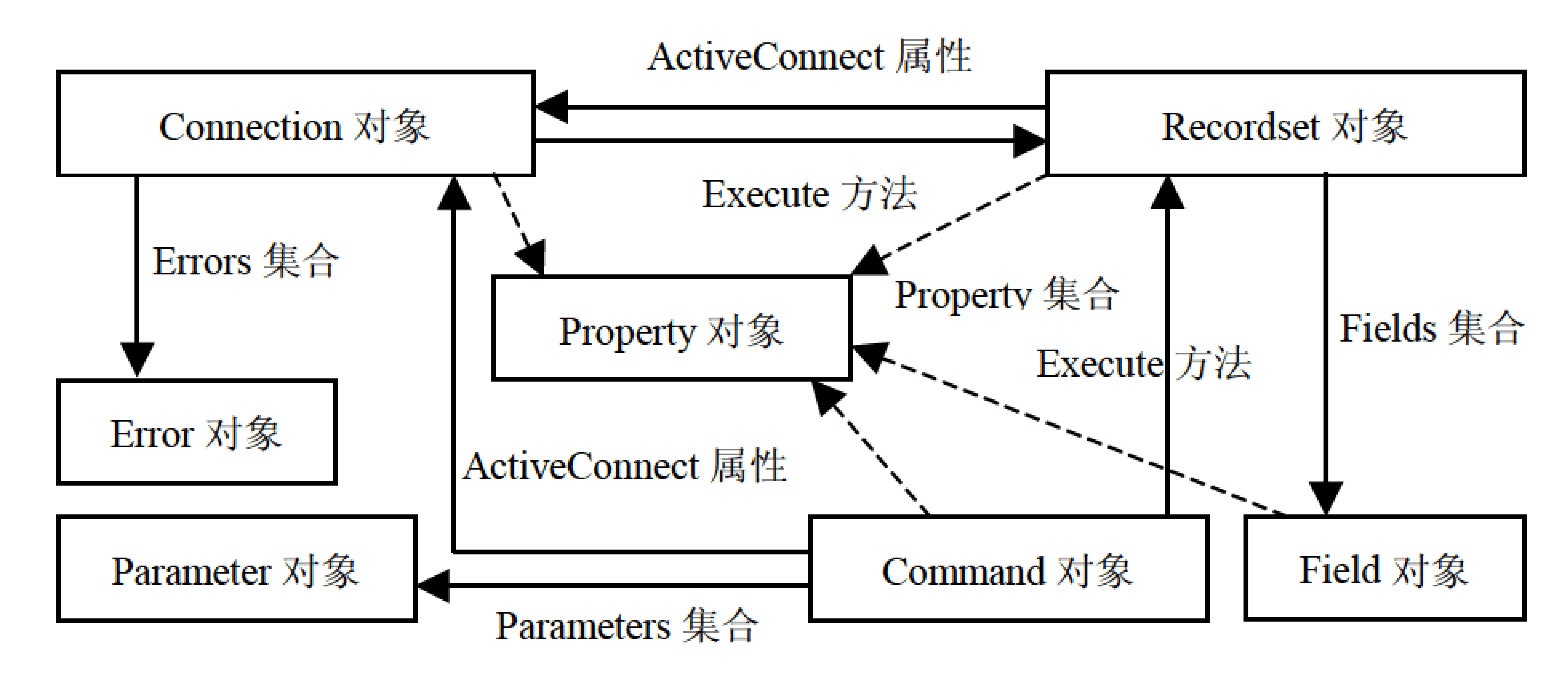


图 5-2　ADO 对象与数据集合之间的关系

5.3　用 Connection 对象连接数据库

ASP 程序要对数据库进行操作，首先要与数据库建立连接。Connection 对象不仅可以实现数据库的连接，而且还可以通过执行 SQL 语句对所连接的数据库进行各种各样的操作。

使用 Connection 对象，应用如下方法创建对象实例：

```
<%  Set conn = Server.CreateObject("ADODB.Connection")  %>
```

5.3.1　Connection 对象的常用属性和方法

1. Connection 对象的方法

（1）Open 方法：用于建立到数据库的物理连接。

（2）Execute 方法：用于执行指定的查询、SQL 语句、存储过程或特定提供者的文本。

（3）Close 方法：关闭 Connection 对象以便释放所有关联的系统资源。需要说明的是，关闭对象并非将它从内存中删除，此时可以更改它的属性设置并在以后再次使用 Open 方法时打开它。若想将对象完全从内存中删除，可以将对象变量设置为 nothing。

2. Connection 对象的常用属性

（1）Provider 属性

Provider 属性用于指定 OLE DB 提供者，即数据访问驱动程序的名称，默认值是 MSDASQL（Microsoft OLE DB Provider for ODBC）。如果 ConnectionString 中未指定 OLE DB 提供者的名称，则使用 Provider 属性指定的名称。

例如，以下使用 OLE DB 连接 Access 数据库。

```
<%
Set conn = Server.CreateObject("ADODB.Connection")
Conn.Provider= "Microsoft.Jet.OLEDB.4.0"
Conn.Open Server.MapPath("user.mdb")
%>
```

（2）ConnectionString 属性

该属性包含用于建立连接数据源的信息。例如，用 ASP 访问 Access 数据库时，可以在连接字符串中包含 DRIVER 和 DBQ 两个参数，分别指定所用的数据库驱动程序和要连接的 Access 数据库文件的路径。例如：

```
<% conn.ConnectionString = "DRIVER = {Microsoft Access Driver (*.mdb)};DBQ = c:\test.mdb" %>
```

（3）ConnectionTimeout 属性

该属性用于设置建立数据库连接期间的等待时间，其为长整型值（单位为秒），默认值为 15。

（4）Mode 属性

该属性用于设置或返回当前连接上提供者正在使用的对数据源的访问权限。其属性值是系统定义的一些常量，只能在关闭 Connection 对象时才能设置。Mode 属性取值如表 5-2 所示。

表 5-2　Connection 对象的 Mode 属性取值

常　　量	取　　值	说　　明
AdModelUnknown	0	默认值，表明权限未设置
AdModelRead	1	表明权限为只读
AdModelWrite	2	表明权限为只写

续表

常　　量	取　　值	说　　明
AdModelReadWrite	3	表明权限为读/写
AdModelShareDenyRead	4	防止其他用户使用读权限打开连接
AdModelShareDenyWrite	8	防止其他用户使用写权限打开连接
AdModelShareExclusive	12	以独占方式连接数据源
AdModelShareDenyNone	16	防止其他用户使用任何权限打开连接

（5）State 属性

该属性用于检查连接是处于打开状态（值为 1）还是处于关闭状态（值为 0）。

5.3.2　连接数据库

使用 Connection 对象的 Open 方法实现与数据库的连接。其语法格式如下：

Open ConnectionString , UserID , Password , Options

其中：

- ConnectionString 为包含连接信息的字符串。
- UserID 为可选项，用于指定建立连接时所使用的数据库的用户名。
- Password 为可选项，用于指定建立连接时所使用的数据库的用户密码。
- Options 为可选项，用于指定打开连接的方式是同步还是异步。默认是同步打开连接，当该选项的值为 adAsyncConnect 时，代表异步连接。异步连接不必等 Open 方法执行完成就立即返回执行下一条语句。通常不使用该选项。

以下将结合 Access 数据库和 SQL Server 数据库介绍具体的连接方法。

1．利用 ODBC 连接数据库

用 ODBC 连接数据库有使用 ODBC 数据源和使用 ODBC 连接字符串两种形式。

（1）利用 ODBC 数据源连接数据库

对于在“ODBC 数据源管理器”设置过的数据源，可以简单地使用如下格式建立连接。

conn.open "数据源名称"

这里，数据源为 ODBC 系统数据源。数据库可以是任何支持 ODBC 连接的数据库。

另外，也可以使用“DSN=数据源名称”的格式进行详细指定，并可根据具体数据库类型省略后面的若干参数。例如：

```
conn.open "DSN=test;UID=sa;PWD=;Database=student"
```

（2）利用 ODBC 连接字符串连接数据库

对于没有在“ODBC 数据源管理器”设置过的数据源，可以在连接字符串中直接指定数据库的 ODBC 专用驱动程序。

① 利用 ODBC 连接字符串连接 Access 数据库

```
<%
Set  conn=Server.CreateObject("ADODB.Connection")
DBPath=Server.MapPath("student.mdb")
conn.Open "Driver={Microsoft Access Driver (*.mdb)};DBQ="&DBPath
%>
```

其中，连接串中各个参数必须使用分隔符“;”（分号）分开。Driver 参数用来指定所要连接数据库的驱动程序，并且必须和指定的数据库为同一类型，该参数的设定需使用{}括住。DBQ 参数用来设置要连接到的数据库的实际路径文件名称。为了考虑应用的通用性，经常将数据库文件与 ASP 程序文件放在同一目录或某个相对子目录下。Server.MapPath 将以 ASP 文件的目录路径为基础，获得要访问数据库文件的物理路径。这里，数据库 student.mdb 与 asp 脚本代码在同一目录下，否则，在 MapPath 的参数中要指定相对路径。

【注意】{Microsoft Access Driver (*.mdb)}语句中的 Microsoft、Access、Driver 以及(*.mdb)之间均只能有一个空格。

② 利用 ODBC 连接字符串连接 SQL Server 数据库

以下设置 Connection 对象的连接字符串，然后用 Open 方法连接 SQL Server 数据库。

```
conn.ConnectionString = "DRIVER={SQL Server}; Server=localhost;" &
  " UID=sa;PWD=password;DATABASE=student"
conn.Open
```

其中，DRIVER 参数指定所用的 ODBC 驱动程序，Server 指定服务器名称，UID 和 PWD 给出登录 SQL Server 数据库服务器的用户名和密码，DATABASE 指定要连接的数据库。

【应用提示】使用 ODBC 连接字符串的优点是不用设置 ODBC，系统使用配置可简化，但其缺点是代码中会暴露数据库的详细信息，如果被非法用户获取，则会给数据库的安全带来隐患。

2．利用 OLE DB 连接数据库

OLE DB 是微软用来替代 ODBC 的一种数据库访问技术。它对关系型数据库和非关系型数据库均适用。以下分别给出用 OLE DB 连接 Access 数据库和 SQL Server 数据库的实例。

（1）用 OLE DB 连接 Access 数据库

```
strconn = "provider=microsoft.jet.oledb.4.0;data source=" & server.MapPath("/database/教学管理.mdb")
conn.open strconn
```

其中：

- provider 指定用于连接的提供者的名称，Access 使用 Microsoft.Jet.OLEDB.4.0。
- data source 指定 Access 数据库的物理位置。

（2）用 OLE DB 连接 SQL Server 数据库

```
set  Conn=Server.CreateObject("ADODB.Connection")
Strcnn="Provider=sqloledb;User ID=sa;Password=1234;Initial Catalog=student;Data Source=PC1"
Conn.open  Strcnn
```

其中：

- ❑ Provider 指定的是采用何种数据库引擎。此处指定 sqloledb 说明用 OLE DB 连接 SQL Server 数据库。
- ❑ User ID 指定的是访问 SQL Server 的用户名。
- ❑ Password 指定 User ID 指定的用户访问密码。
- ❑ Initial Catalog 指定要访问的数据库名称。
- ❑ Data Source 指定要访问的 SQL Server 服务器的计算机名，此处指定计算机名为 PC1，也可以将其值设置为 IP 地址。

5.3.3　用 Connection 对象执行 SQL 语句

利用 Connection 对象的 Execute 方法执行 SQL 语句实现对数据库表格的各类访问处理。

Execute 方法的调用分为有括号和无括号两种形式，前者用于有记录集返回的 SQL 语句（如 select 语句），后者则执行一个无返回结果的 SQL 命令（如 insert 语句）。

1．执行 select 语句返回记录集对象

格式：Set rs = Conn.Execute(sql)

其中，sql 字符串为一个 select 查询语句。

2．无记录集返回的命令执行（如 insert、update、delete 等 SQL 命令）

格式：Conn.execute"SQL 操作性语句"[,RecordAffected][,Option]

其中：

- ❑ CommandText 是一个字符串，包含要执行的 SQL 语句、表名、存储过程或特定提供程序的文本。
- ❑ RecordAffected 为长整型变量，返回操作所影响的记录数目。
- ❑ Option 指定 CommandText 字符串的命令类型，常用取值及含义如表 5-3 所示。

表 5-3　Option 选项的取值

常　　量	取　　值	说　　明
adCmdText	1	表明 CommandText 包含一个命令文本定义
adCmdTable	2	表明 CommandText 为一个表的名字
adCmdStoredProc	4	表明 CommandText 为一个存储过程名
adCmdUnknown	8	指示 CommandText 参数中的命令类型为未知（默认）

【例 5-1】　用户注册处理程序

程序 1：注册填写程序

表单部分程序的主要代码如下：

```
------------------------------------------------------ex5-1.htm--------------------------------------------------------
<form action="wishuserprocess.asp" method="post">
<TABLE  border="1"   cellspacing="0"  cellpadding="0"  id="table1"  style="width:  336px"  bgcolor=
```

```
"#F5FEFE">
<TR><TD align=center width="82%" bgColor=#C5FCF5 colspan="2" height=28>
  <p  align="left">   <img  border="0"  src="images/icon_readmore.gif"  width="15"  height="12">
<font color="#008080">注册用户请真实填写以下内容，以便联系</font></TD></TR>
<TR><TD align="left" width="33%"   ><font   color="#000000">  
     用户名</font><b><font   color="#FF0000"> * </font></b> </TD>
<TD align=left   width="62%" valign="top">
 <input name="no" type="text" size="20"></TD> </TR>
<TR> <TD align="left" width="33%"   ><font   color="#000000">  
     密  码 </font><b><font color="#FF0000">*</font></b></TD>
<TD align=left width="62%" valign="top">
<INPUT name="password" type="password" size="21">
</TD> </TR>
<TR> <TD align="left" width="33%"   ><font   color="#000000">  
     确认密码 </font><font color="#FF0000">*</font></TD>
<TD align=left width="62%">
<INPUT name="confirm" type="password" size="21">
</TD> </TR>
<tr><TD align="left" width="33%"   ><font   color="#000000">  
     姓  名 </font><b><font   color="#FF0000">*</font></b></TD>
<TD align=left   width="62%">
<INPUT type="text" size="20" name="user" >
  </TD> </tr>
<TR> ……</TR>
</TABLE>
<p align="center"><br><INPUT type="submit" class=button value="注   册  "> </p>
</form>
```

【说明】该程序提供了一个注册表单填写页面，通过表格可实现输入提示及表单各个输入域的排列定位。如图 5-3 所示为新用户注册界面。

图 5-3　新用户注册界面

程序 2：注册处理程序

```
---------------------------------------------------wishuserprocess.asp------------------------------------------------
<%
file="data.mdb"
str="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Set Conn=Server.CreateObject("ADODB.connection")
```

```
Conn.Open str
sql="select * from passwd where userid='"&Request.Form("no")&"'"
set rs=conn.execute(sql)
if not rs.eof   then %>
    <center><br><br><font size=5> 该用户已注册！ </font></center>
Else
    sql="insert into passwd values('"&trim(request("no"))&"','"
      &request("password")&"','"&request("user")&"','"&request("address")
      &"','"&request("email")&"',dateValue('"&date()&"'))"
  conn.execute(sql)
%>
  <center><br><br><br> <font size=5> 注册成功！ </font></center>
<%end if%>
```

【说明】首先执行查询语句检查用户是否已注册，如果账户已注册，则输出提示。对于未注册用户，则通过执行 insert 语句将新用户写入数据库表格。使用&运算符进行 SQL 字符串的拼接，要特别注意使 SQL 串中数据的类型与数据库表格的字段类型保持一致。在串的拼接中注意单引号和双引号的使用，双引号表示 VBScript 字符串的开始与结束，而单引号是 SQL 命令中字符串数据的定界符。另外还应注意，日期型数据是利用 SQL 命令的 dateValue 函数转换实现的。

【注意】对 Access 数据库的写入还涉及权限问题，要保证数据库所在目录和数据库文件对 IIS 的匿名账户是可写的。

5.3.4　Connection 对象的数据集合

Connection 对象有 Errors 和 Properties 两个数据集合，前者表示 Connection 对象运行时最近一次出现的错误信息，后者表示 Connection 对象的相关属性的集合。

1．Error 对象与 Errors 数据集合

（1）Error 对象

每个 Error 对象代表了特定数据提供的错误，利用其属性可以得到错误的相关信息，如 Number 为错误编号，Description 为错误描述等。

（2）Errors 数据集合

Errors 数据集合是当前连接进行数据访问操作所发生的所有错误的集合，Errors 的属性和方法如下。

- Count 属性：集合中所包含的 Error 对象的个数。
- Clear()方法：清除集合中的元素。
- Item(i)方法：获取集合中序号为 i 的 Error 对象。其中，i 的取值从 0 开始。

2．Property 对象和 Properties 数据集合

ADO 的各个对象都有很多属性，每个属性都是一个 Property 对象，它们拥有各自的数据类型、名称和值等。为了方便控制，将同一 ADO 对象的属性汇集在 Properties 集合中。

（1）Property 对象

Property 对象代表由数据提供者定义的 ADO 对象的动态特性。ADO 对象有两种类型的属性，即内置属性和动态属性。内置属性是 ADO 对象的固有属性，不会作为 Property 对象出现在 Properties 集合中；动态属性随程序的运行由数据提供者定义，将作为 Property 对象出现在 Properties 集合中，注意，它只能通过 Properties 集合的属性和方法引用。Property 对象的属性如表 5-4 所示。

表 5-4　Property 对象的属性

属　　性	说　　明
Name	Property 对象的名称
Value	Property 对象的值
Type	Property 对象的数据类型。对于 Parameter 对象，该属性可读写；对于其他 ADO 对象，该属性只读
Attributes	Property 对象的特性码

（2）Properties 集合

Properties 集合对象中包括了一个 ADO 对象实例的所有 Property 对象，Properties 集合的属性和方法如表 5-5 所示。

表 5-5　Properties 集合的属性和方法

属性或方法	说　　明
Count 属性	Properties 集合中 Property 对象的个数
Refresh 方法	刷新集合，重新取得集合中所有 Property 对象
Item 方法	取得集合中某个 Property 对象

【例 5-2】　访问 connection 实例的各 Property 对象的信息

```
-------------------------------------------------------ex5-2.asp-------------------------------------------------------
<%
Set Conn = Server.CreateObject("ADODB.connection")
Conn.Open "studentscore"      '用 ODBC 数据源连接数据库
Response.Write "<table border=1 ><td >属性名</td><td >属性类型</td>"
Response.Write "<td >属性值</td><td >特性</td>"
for each prop in Conn.Properties
   Response.Write "<tr><td>" & prop.name &"</td>"
   Response.Write "<td>" & prop.type &"</td>"
   Response.Write "<td>" & prop.value &"</td>"
   Response.Write "<td>" & prop.attributes &"</td></tr>"
next
%>
```

5.3.5　Connection 对象的事务处理

事务是指一组相关操作要么全部成功，要么全部取消。Connection 对象的一个重要功

能是执行并控制数据源的事务操作。例如，在网上教学系统中，需要根据班级学生名单批量建立账户，这里需要给每个账户指定一个登录名，可以全班统一用相同前缀并在后面加学生序号来代表登录名，但在添加用户的操作过程中可能有某个账户已被注册过，则说明该班的账户前缀选得不合适，整个添加操作全部取消。

1．事务处理方法

事务处理包含以下 3 个方法。

（1）BeginTrans 方法：标识事务的开始。

（2）CommitTrans 方法：事务提交。

（3）RollbackTrans 方法：事务回滚。

使用事务可以优化写操作的过程，从事务开始对数据源的写操作是在内存缓冲区进行，事务提交时才把缓冲区内容写入到数据库。

2．ASP 的错误处理

在编写事务处理代码时，常需要结合使用 ASP 的错误处理机制。ASP 错误处理的思想是，让程序遇到错误时继续执行，在特定的执行点再通过 Err 类的属性和方法来检查和处理错误。为了让程序遇到错误时继续执行，在可能出错的代码之前加上如下语句：

```
On Error Resume Next
```

要判断是否有错误发生，可经常使用 Err 类的 Number 属性，当程序没有错误产生时其值为 0。Err 类的属性和方法如表 5-6 所示。

表 5-6　Err 类的属性和方法

属性	Number	错误编号
	Description	错误的简短说明
	Source	错误的对象源，应用程序的程序设计 ID
方法	Raise	强制产生错误，如 Err.raise 6 产生溢出错误
	Clear	清除错误，用于显式地重新设置 Err

【例 5-3】　错误处理测试

```
------------------------------------------------------ex5-3.asp------------------------------------------------------
<%
Response.Buffer = True
On Error Resume Next
x=5/0
If Err.Number <> 0 Then    %>
  错误 Number：  <%= Err.Number %><BR>
  错误信息：  <%= Err.Description %><BR>
  出错文件：  <%= Err.Source %><BR>
<%End If%>
```

访问运行程序将产生如下输出：

错误 Number：11

错误信息：被零除
出错文件：Microsoft VBScript 运行时错误

5.4　用 Recordset 对象访问数据库

Recordset 对象是 ADO 对象中使用最灵活、功能最强大的对象，利用该对象几乎可以完成对数据库的所有访问处理操作。Recordset 对象表示的是来自基本表或命令执行结果的记录集合。该集合就像一个二维数组，数组的每一行代表一条记录，而每一列代表数据库表格的字段，并用记录指针指示当前的那条记录。

5.4.1　Recordset 对象的创建

一般来说，有两种方法创建记录集对象实例。

（1）利用 Connection 或 Command 对象的 Execute 方法从一个数据库返回结果时将自动创建一个记录集对象实例。

（2）通过 Server.CreateObject("adodb.recordset")创建 Recordset 对象，然后通过该对象的 Open 方法创建一个与数据库表格关联的 Recordset 对象实例。例如：

```
strSQL="select * from student"
Set rs=Server.CreateObject("adodb.recordset")
rs.open strSQL,Conn,1,1
```

该方法所获得的记录集具有更灵活的控制性和更强的功能。在打开记录集之前，可以详细设置记录集的游标和锁定类型。Recordset 对象的 Open 方法的语法格式如下：

open usersql,myconn,cursortype,locktype,option

其中：

- usersql 代表 SQL 查询语句。
- myconn 代表所依赖的 Connection 对象的实例。
- cursortype 用于设置记录集的游标类型，从而可以决定对记录集进行怎样的操作。该选项具体取值如表 5-7 所示。

表 5-7　记录集的游标类型

符号常量	值	功能描述
adOpenForwardOnly	0	前项游标，默认值。只能向前移动
AdOpenKeyset	1	可向前或向后移动游标，当其他用户删除或改变一条记录后，记录集将反映出这个变化，若用户添加新记录，该新记录将不会出现在记录集中
AdOpenDynamic	2	可向前或向后移动游标，其他用户的任何修改都将在记录集中立即反映出来
adOpenStatic	3	静态游标。其他用户的任何修改都不在记录集中反映出来；服务器响应的数据与数据库已经分开

【注意】VBScript 中定义的符号常量文件位于 C:\Program Files\Common Files\System\ado\adovbs.inc，如果应用中需要用到其中的常量，可以将该文件复制到 ASP 应用的目录下，在需要用到符号常量的程序前加上如下语句：

```
<!-- #include file = "adovbs.inc" -->
```

- ❑ Locktype 用于设置对记录集的锁定类型，它决定了多用户试图同时修改一条记录时对记录应如何处理。该选项具体取值如表 5-8 所示。

表 5-8　记录集的锁定类型

符 号 常 量	值	功 能 描 述
adLockReadOnly	1	以只读方式打开，所以不能做任何更新；为默认值
adLockPressimistic	2	悲观加锁，在编辑修改记录时，立即锁定它
adLockOptimistic	3	乐观加锁，在编辑修改记录时，并未加锁，只有在调用记录集的 Update 方法更新记录时，才锁定记录

- ❑ Option 用于指定 usersql 参数项的命令字符串的类型。该选项具体取值如表 5-3 所示。

5.4.2　记录集游标及移动方法

对记录集的访问是逐个记录进行的，每次操作访问的记录称为当前记录。所谓记录集游标就是指当前记录指针。通过移动游标可以实现对整个记录集中各记录的遍历访问。记录之间的前后关系及记录指针的移动方向关系如图 5-4 所示。

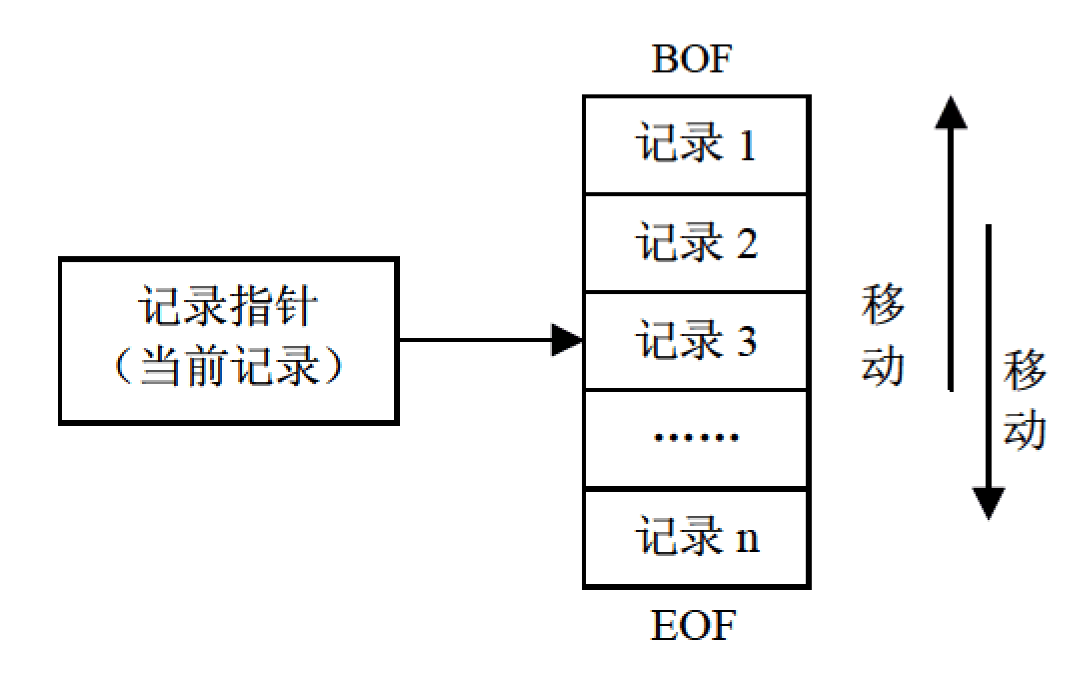


图 5-4　记录指针移动方向示意图

1．Move 方法

该方法将记录指针向前或向后移动指定的记录个数。其语法格式如下：

Move NumRecords, Start

其中：

- ❑ NumRecords 表示指针移动的记录数目。当为正数时表示向后移动；当为负数时表示向前移动。
- ❑ Start 表示指针移动的基准点。该选项的取值如表 5-9 所示。默认是根据记录集的 Bookmark（书签）当前值进行相对移动。读记录集的 Bookmark 属性得到的是当前记录位置，以下代码将 Bookmark 属性保存在一个变量中，以后可以简单地通过设置 Bookmark 属性的方法回到该记录，从而实现指针的快速移动。

```
Mark = rs.Bookmark          '记住位置
……                          '移动记录到别的位置
rs.Bookmark = Mark          '改变书签到以前记住的位置
```

表 5-9　指针移动的基准点

常　　量	常　量　值	说　　明
AdBookmarkCurrent	0	根据书签的当前值进行相对移动
AdBookmarkFirst	1	从首记录开始
AdBookmarkLast	2	从尾记录开始

2．MoveFirst、MoveLast、MoveNext 和 MovePrevious 方法

（1）MoveFirst：将记录指针移至第一条记录。

（2）MoveLast：将记录指针移至最后一条记录。如果记录集不允许使用书签，则该方法会产生错误。

（3）MoveNext：将记录指针移至下一条记录。如果 EOF 为 True，则该方法会产生错误。

（4）MovePrevious：将记录指针移至上一条记录。如果 BOF 为 True，则该方法会产生错误。

3．BOF 与 EOF 属性

（1）BOF 属性：若当前记录的位置在第一行记录之前，则 BOF 属性返回 True，反之则返回 False。

（2）EOF 属性：若当前记录的位置在最后一行记录之后，则 EOF 属性返回 True，反之则返回 False。

当对记录集进行遍历访问时，经常要使用 BOF 或 EOF 属性判断记录指针是否到达记录集的边界位置。如果 BOF 与 EOF 都为 True，则表示在记录集中无任何记录。

5.4.3　访问记录的数据内容

一条记录的内容是若干字段（Field）的数据构成的集合（Fields），读记录数据就是对这些字段的访问。

1．Fields 集合

Fields 集合最常用的属性是 Count，它可以获得集合中所包含的 Field 对象的个数，也就是记录集的字段数。

Item 方法是 Fields 集合的常用方法，用于获得集合中的某一 Field 对象。其语法格式如下：

Rs.Fields.item(index)

其中，index 既可以是字符串形式的字段名，也可以是一个代表字段序号的值，序号从 0 开始编号。也就是说，对字段的访问既可通过字段名，又可通过字段的顺序号实现。例如，要访问记录集中字段名为 username 的字段，并假设其序号为 0，则有如下几种表达形式：

- rs("username ")
- rs(0)
- rs.Fields("username ")
- rs.Fields(0)

- rs.Fields.item("username ")
- rs.Fields.item(0)

2．Field 对象

每个 Field 对象对应记录集的一个字段（列），其常用属性有以下几种。

- Name 属性：字段名。
- Type 属性：字段类型。
- Value 属性：字段值。
- ActualSize 属性：字段的实际长度。
- DefineSize 属性：字段在数据库中定义的长度。

如果访问某字段不指定属性，则默认指访问字段的值（Value 属性），以下的表达形式等价：

```
rs.Fields.item("username")
rs.Fields.item("username").value
```

访问字段的其他属性要具体指定，如 rs(i).name 表示获取序号为 i 的字段的名称。要将记录集的所有字段名显示出来，可用如下循环实现。

```
For k=0  to  rs.Fields.count-1
   Response.Write rs(k).name & ","
Next
```

【思考】如何用 for each…next 循环输出 Fields 集合中的所有字段名。

5.4.4　记录集的分页显示

前面介绍了如何检索数据并输出到浏览器端，对少量数据而言，简单的输出处理是完全可以的，但是若数据量很大，有几百条甚至上千条，一次将如此多的数据全部输出到客户端是不现实的，一是页面从上到下拉得很长，二是客户端等待的时间过长，三是服务器的负载过大，所以应经常采取分页输出。与记录集的分页处理相关的几个属性如表 5-10 所示。

表 5-10　与记录集的分页处理相关的几个属性

属　　性	功　　能
AbsolutePage	设定当前记录的位置是位于哪一页，以 1 为起始页
AbsolutePosition	目前记录指针在 Recordset 中的位置
PageCount	Recordset 对象包括多少“页”的数据
PageSize	Recordset 对象每一页显示的记录数
RecordCount	Recordset 对象记录的总数

处理步骤：首先用 PageSize 指定页的大小。在此基础上，用 PageCount 得到页面的总数。通过 AbsolutePage 属性定位到指定页，从而对该页数据进行访问。

【例 5-4】　网络教学系统用户留言的分页查询

在网络教学平台的首页安排了用户留言的超链接，此时涉及了一个 guestbook 表格，该表的字段说明如表 5-11 所示。

表 5-11　guestbook 表的字段说明

字　段	类　型	意　义
id	自动类型	用于唯一性标识问题
title	备注	来宾留言的内容
flag_answer	是/否	是否回答
answer	备注	管理员解答
time_qry	日期/时间	提交日期/时间

以下是与翻页处理相关的程序代码：

```
----------------------------------------------------guestbook.asp----------------------------------------------------
<%
strPage = Request.queryString("page")
file="data.mdb"
Set objCnn = Server.CreateObject("ADODB.connection")
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
objCnn.Open connstr
strSQL = "select * from guestbook order by time_qry desc"
set rs=server.createobject("adodb.recordset")
rs.open strSQL,objCnn,1,1
if rs.RecordCount>0 then
rs.PageSize = 10
intPageCount = rs.PageCount
if Len(strPage) = 0 then strPage = "1"
if int(strPage)<1 then strPage = "1"
if int(strPage)>intPageCount then strPage = CStr(intPageCount)
rs.absolutepage=cint(strpage)
%>
<form name="frmSearch" action="guestbook.asp" method="get" >
<table width="96%" cellpadding="5" cellspacing="0" >
<tr>
<td align=left>
【<a href="input_guestbook.asp"><font color=green>我留言</font></a>】   
[本页是第<%=int(strPage)%>/<%=intPageCount%>页]
</td>
<td align=right>第<input id="page" name="page" type="text" style="width: 20px;" value="<%=
strPage%>" /> 页
<input type='image' src="images/btn_go.gif">
<a href="guestbook.asp?page=1"> 第一页</a>
<a href="guestbook.asp?page=<%=CStr( int(strPage)-1 )%>"> 上一页</a>
<a href="guestbook.asp?page=<%=CStr( int(strPage)+1 )%>">下一页</a>
<a href="guestbook.asp?page=<%=intPageCount%>">最后页</a>
</td>
</tr>
```

```
</table><br>
<table width="98%" style='table-layout:fixed' >
<tr bgcolor="#EBFED1" height="25">
<td width="5%" align=center><font color="#008000">序号</font></td>
<td width="80%" align=center><font color="#008000">留言内容和应答</font>
</td>
<td width="10%" align=center><font color="#008000">留言时间</font></td>
</tr>
<%
dim flagcolor,classtype
flagcolor = true
for num = 1 To rs.PageSize
if rs.EOF then
	exit for
end if
flagcolor = not (flagcolor)
if (flagcolor) then
	classtype = "tr1"	'用不同的样式交替显示各行
else
	classtype = "tr2"
end if
%>
<tr class="<%=classtype%>">
<td align=center><%=num%></td>
<td style="word-wrap:break-word">
<pre><%If rs("title")<>"" Then
	Response.Write Server.htmlEncode(rs("title"))
	End if%></pre>
<%
	if (rs("flag_answer")) then
		Response.Write "<pre><font color=#993399><b>答:</b></font>"
			&server.htmlencode(rs("answer"))&"</pre>"
	else
		Response.Write "<font color=#993399><b>未回答</b></font>"
	end if
%>
</td>
<td align="center"><%=rs("time_qry")%> </td>
</tr>
<%
rs.MoveNext
Next
objCnn.close
%>
</table>
</form>
<% end if%>
```

运行结果如图 5-5 所示。



图 5-5　分页查看用户留言界面

【说明】本例主要介绍分页显示查询的实现思想，翻页超链接和表单提交的目标程序均为同一个程序文件。在这里，获取当前页用 Request.queryString("page")，所以表单的提交方式要使用 get 方式。为了让各行内容更加醒目，此时使用不同的背景样式交替显示各行。

5.4.5　记录的添加与编辑修改

利用记录集提供的一组方法，可以实现记录的添加、更新、删除操作。

1．AddNew 方法

该方法用于向记录集中插入一条新记录，使用本方法后，新插入的记录将成为当前记录。其语法格式如下：

rs.AddNew FieldList, Values

其中：

- ❑ FieldList：可选参数，新记录中字段名列表。
- ❑ Values：可选参数，与 Field 对应的一组字段值。

通常采用如下形式插入新记录：

```
rs.addnew
rs("字段名 1")=值 1                    '为字段赋值
rs("字段名 2")=值 2
rs.update
```

在立即更新模式下，调用不带参数的 AddNew 方法，字段值的更改缓存在本地，调用 Update 方法时才将新记录传递到数据库，只有这样，才能在进行多记录操作时提高应用性能。

【应用实例】在网络考试交卷时，要将学生解答写入数据库，可以使用循环将所有解答添加到记录集中，然后在循环外使用 Update 方法更新数据。在应用中由于要多次访问到试题和学生解答，所以事先将数据存储在数组中，以提高性能。

```
i=1
do while i<=amount                  'amount 为试题数量
      rs.addNew
      rs("xuehao") = username       '学生标识
      rs("tihao") = shiti(i)        '第 i 道题的试题编号
      rs("answer") = answer(i)      '第 i 道题的试题的学生解答
      rs("type") = 1                '1 代表单选题
      i=i+1
loop
rs.update                           '最后一次性更新数据库表格
```

【例 5-5】　批量建立学生用户

全班用户的登录名使用统一前缀，批量建立账户，数据库表中用户登录名的值具有唯一性，如果出现重名，则全班操作取消。

以下处理程序需要用到请求页面中表单提交的 3 个域变量。

- classname：班级名。
- students：班级学生名单清单，各学生之间用逗号分开。
- prefix：用户登录名的前缀。

```
-------------------------------------------------------ex5-5.asp-------------------------------------------------------
<%
'本程序演示 ASP 事务处理的应用
on error resume next
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open request.cookies("connstring")        '建立数据连接
Conn.begintrans                                '事务开始
Set rs=Server.CreateObject("ADODB.Recordset")
sql="SELECT * FROM   usertable "
rs.open sql ,conn,1,2                          '建立记录集
prefix =request("prefix")
classname=request("classname")
alluser=request("students")
user=split(alluser,",")                        '将学生名单分离存储在数组 user 中
amount=ubound(user)+1                          '求学生用户数量
for n=1 to amount
   rs.addnew                                   '添加一个记录
   rs("loginid")= prefix &n                    '第 n 个学生的登录名
   rs("password")="111"                        '初始密码统一为 111
   rs("username")= user(n-1)                   '第 n 个学生的姓名
   rs("classname")= classname
next
rs.update                                      '更新记录
if   Err.Number=0 then                         '说明无错产生
   conn.CommitTrans                            '事务提交
   response.write "<font color=red>导入成功,祝贺!</font>"
else
   conn.RollbackTrans                          '事务回滚
   Response.Write "<font color=red>导入失败，系统已有同名账户!</font>"
end if
```

```
conn.close
set conn=nothing
%>
```

【说明】本例用到了接下来将要介绍的记录集对象实现对数据库表格的操作。利用记录集的 addNew 方法添加一条记录，并通过 “rs("字段名")=值” 的方式为字段赋值。循环后通过 update 方法实现操作更新。程序中利用字符串的 split 方法将学生名单存储到数组中，以便进行操作访问。在整个程序的最后安排了错误判定代码，通过 Err 类的 Number 属性判断是否出现错误，如果有错，则执行 Connection 对象的 RollbackTrans 方法将事务回滚；否则，执行 CommitTrans 方法实现事务提交。

2．Delete 方法

该方法用于删除当前记录或一组记录。其语法格式如下：

rs.Delete AffectRecords

其中，AffectRecords 为可选项，确定 Delete 方法所影响的记录数目，具体常量定义如表 5-12 所示。

表 5-12　AffectRecords 的取值

常　　量	说　　明
AdAffectCurrent	默认。仅删除当前记录
AdAffectGroup	删除满足当前 Filter 属性设置的记录。要使用该选项，必须将 Filter 属性设置为有效的预定义常量之一
adAffectAll	删除所有记录
adAffectAllChapters	删除所有子集记录

使用立即更新模式将在数据库中进行立即删除，否则记录将标记为从缓存删除，实际的删除处理将在调用 Update 方法时进行。

3．Update 方法

该方法将 Recordset 对象的当前记录所做的所有更改保存到数据库中。其语法格式如下：

rs.Update Fields, Values

其中：

- Fields：可选参数，要修改的字段名列表。
- Values：可选参数，与 Fields 对应的一组字段值。

【例 5-6】　试题库中单选题的管理

单选题的数据库表格 single_select 的字段说明如表 5-13 所示。

表 5-13　single_select 表的字段说明

字　　段	类　　型	意　　义
st_no	自动类型	试题编号
content	备注	试题内容
answer	文本	标准答案

续表

字　　段	类　　型	意　　义
difficulty	数字	难度系数
point_no	文本	所属知识点

程序 1：分页查看试题

```
----------------------------------------------------view_select.asp----------------------------------------------------
<%
file="data.mdb"
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open connstr
Set rs=Server.CreateObject("ADODB.Recordset")
sql="SELECT * FROM single_select "
rs.open sql,conn ,1,2
mlngTotalPgs = rs.RecordCount
if mlngTotalPgs<>0 then
    mlngTotalRds = mlngTotalPgs
    mlngTotalPgs = int((mlngTotalPgs+10-1)/10)
else
    mlngTotalRds = mlngTotalPgs
end if
iCurPage=clng(request("page"))
if iCurPage<1 then iCurPage=1
if mlngTotalPgs<>0 then
    rs.move (iCurPage-1)*10
end if %>
<html>
<head><title>试题查询</title>
<link rel="stylesheet" type="text/css" href="main.css">
<script language="javascript">
function gotopage(i) {
      frmSearch.page.value = i;        //在 page 框中设置页码
      frmSearch.submit();              //提交表单
}
</script>
</head>
<body><center>
<form name="frmSearch" action="view_select.asp" method="get" >
<div align="center" class="title">
<font color="#339966">单选题管理</font></div>
<!--   以下表格显示当前页及记录的信息以及翻动试题的超链接 -->
<table border="1" width="95%" height="208" cellspacing="0" >
<td height=20 align=left>共有<font color=red><%=mlngTotalRds %></font>
条记录  显示第<font color=red><%=iCurPage%></font>页/共
<font color=red><%=mlngTotalPgs%></font>页
</td>
<td   height=16 align=right>
<%if iCurPage <= 1 then
```

```
        Response.Write "最前页"
elseif iCurPage > 1 then
      Response.Write "<a   href='javascript:gotopage(1);'>"& "最前页" & "</a>"
end if
      Response.Write " "
if iCurPage <= 1 then
      response.write "上一页"
elseif iCurPage > 1 then
      response.write "<a   href='javascript:gotopage("&iCurPage-1&");'>"
            & "上一页" & "</a>"
end if
      response.write " "
if iCurPage >= mlngTotalPgs then
      response.write "下一页"
else
      response.write "<a   href='javascript:gotopage("&iCurPage+1&");'>"
            & "下一页" & "</a>"
end if
      response.write " "
if iCurPage >= mlngTotalPgs then
        response.write "最后页"
else
      response.write "<a   href='javascript: gotopage("&mlngTotalPgs&");'>"
            & "最后页" & "</a>"
end if
response.write "  转向第<input type=text id=""page"" style=
        'WIDTH: 30px' value="&iCurPage&" >页
        <input type='image' src=""go.gif""    id='image1' ></TD>"
%>
<td align="right" width="70">
<input type="button" name="button" value="添加" class="btn"
            onclick="window.location.href='select_insert.asp';">
</td>
</table>
<% if mlngTotalPgs > 0 then %>
<table width="95%" border="1" style="table-layout:fixed">
<!--  以下显示试题内容的表头  -->
<tr height="22" style="background-color: #EBF8CF" valign="bottom" >
<td width="60%" align="middle" ><font color="#D05411">题目内容</font></td>
<td width="5%" align="middle" ><font color="#D05411">答案</font></td>
<td align="middle" width="5%"><font color="#D05411">难度</font></td>
<td width="10%" align="middle" ><font color="#D05411">知识点</font></td>
<td width="10%" align="middle" ><font color="#D05411">操作</font></td>
</tr>
<!--  以下显示试题内容的各行，每页显示 10 条记录  -->
<%
for i=1 to 10
        if rs.Eof then
            exit for
        end if
```

```
%>
<tr>
<td style='word-wrap:break-word'>
<pre><%=server.htmlencode(rs("content"))%> </pre></td>
<td align=middle><%=rs("answer")%></td>
<td align=middle><%=rs("difficulty")%></td>
<td align=middle><%=rs("point_no")%></td>
<td align=center ><a href='select_man.asp?num=<%=rs("st_no")%>
  &ACT=EDIT&page=<%= iCurPage %>'>修改</a>
<font color=green><b>/</b></font>
<a href='select_man.asp?num=<%=rs("st_no")%>&ACT=DEL
&page=<%= iCurPage %>' onclick="return confirm('确定删除吗？ ');">删除</a>
</td>
</tr>
<% rs.movenext
next
%>
</table>
<% end if %>
</form>
</body>
</html>
```

运行结果如图 5-6 所示。

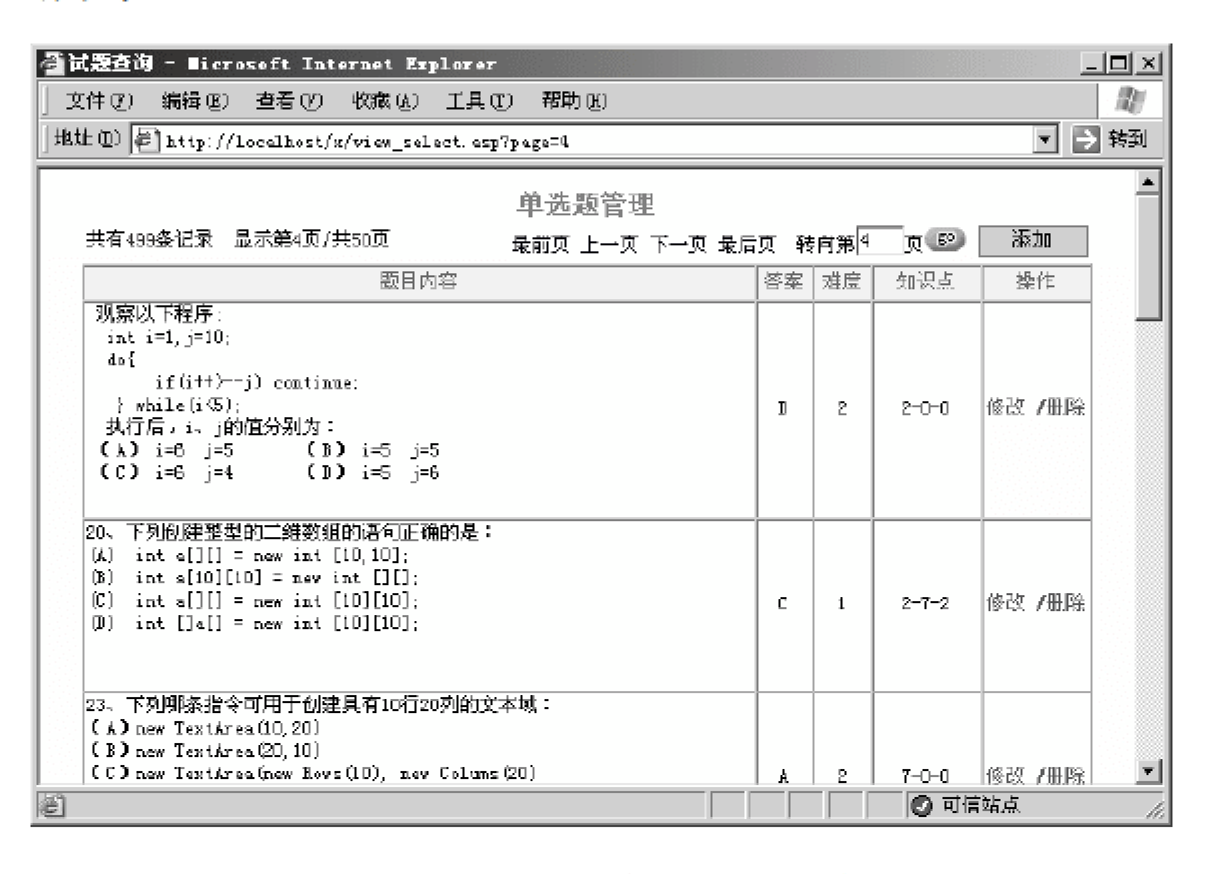


图 5-6　试题的分页浏览

【说明】本例的翻页处理与例 5-4 的方法不同，程序中固定页的大小为 10 条记录，跳转到指定的页是通过记录值的计算将记录移到指定页的开始。翻动页是通过表单提交页码实现，程序中通过 JavaScript 函数 goto 的执行在 page 输入框中设置页码，然后调用表单的 submit()方法提交表单，表单提交的目标页是该页自身。每行提供两个超链接实现该试题的修改和删除操作。

【注意】程序中利用表格和单元格的样式控制固定单元格宽度，使显示内容自动转行。同时通过 server.htmlencode 方法对显示内容进行转换处理，以实现内容的原样显示。

程序 2：修改和删除试题

删除试题比较简单，只需根据 URL 参数提供的试题编号找到试题，用 SQL 中的 delete

命令删除即可。而修改则要提供一个表单，表单中显示试题的原来信息，通过表单提交将新修改的信息写入数据库。

```
--------------------------------------------------select_man.asp--------------------------------------------------
<%
page=clng(request("page"))
sAction = request.querystring("ACT")
file="data.mdb"
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open connstr
strSelNo=request("num")
if sAction ="DEL" then                                    '试题删除处理
   sql="delete FROM single_select where st_no=" &strSelNo
   Conn.Execute(sql)
   Conn.close
   Response.Redirect("view_select.asp?page="&page)
end if
if sAction ="EDIT"   then                                 '试题修改代码
   Set ip=Server.CreateObject("ADODB.Recordset")
   Set cmdTemp=Server.CreateObject("ADODB.Command")
   sql="SELECT * FROM   single_select where st_no="&strSelNo
   ip.open sql ,conn,1,2
   if request("UPD")=1 then                               '判断当前状态是修改显示还是修改数据写入
     ip("content")=request.form("comment")                '将修改结果写入数据库
     ip("answer")=Request.Form("答案")
     ip("difficulty")=request.form("难度系数")
     ip("point_no")=request.form("point")
     ip.update
     ip.close
     Response.Redirect("view_select.asp?page="&page)
   else
     strContent=server.htmlencode(ip("content"))          '对试题内容进行显示并转换处理
     strDiff=ip("difficulty")
     strAnswer = asc(ip("answer"))
     knowledge = ip("point_no")
   end if
%>
<html>
<head>
<link rel="stylesheet" type="text/css" href="main.css">
</head>
<body><br>
<div align="center" class=title>单选题编辑<center><br>
<form name="frmSearch" action="select_man.asp?ACT=<%=sAction%>
   &UPD=1&num=<%=strSelNo%>" method="POST" onsubmit="return check()">
<table border="1" width="95%" height="208" cellspacing="0" >
<tr><td align="center" width="18%" height="95">
<font color="#FF0000">◆</font><font color="#0000FF">试题内容
</font><br><font color=teal>(每行结束请敲回车)</font> </td>
```

```
<td align="left" width="80%" height="95" >
<textarea name="comment" wrap="soft" rows="7" cols="84">
<%=strContent%></textarea> </td>
</tr><tr>
…… 难度的编辑显示省略 ……
<td align="CENTER" width="18%" height="26">
<font color="#FF0000">◆</font><font color="#0000FF">答案</font></font></td>
<td ALIGN="CENTER" width="80%" height="26" >
<table   width="210" align="left">
<%  i=65             '65 为字母 A 的 ASCII 码值
do while i<=69
%>
<td align="center" width="96"> <%=chr(i)%> </td>
<td align="center" width="104">
<input type="radio" size="30" name="答案" value="<%=chr(i)%>"
<%if i=strAnswer then     response.write " checked"
 end if%>></td>
<%
i=i+1
loop
%>
</tr>
</table></td>
</tr> <tr>
…… 知识点的显示与输入省略 ……
</table>
<p><br>
<input type="submit" name="b1" value="确   定" class=btn>
<input type="reset" name="b1"   value="复   位" class=btn>
<input type="button" value="返   回" class=btn onclick="history.back();">
</p>
</form>
</body>
</html>
<%end if%>
```

图 5-7 所示为试题编辑界面。



图 5-7　试题编辑界面

【说明】思考如何将试题编号在页面超链接间进行传递，以便删除和修改完该试题后能显示相应页，让用户清晰地看到操作效果。本例演示了在同一程序文件中实现修改的编辑显示与写入处理，引入了一个 URL 参数 UPD 来区分，在修改写入处理时该参数值为 1。

新试题输入页面与修改页面相同，只是在输入框中均无数据。其写入处理也和修改类似，只是先用 AddNew 方法插入一条新记录，然后将数据写入该记录。读者可自行补充完成相应功能的设计。

4. CancelUpdate 方法

该方法用于取消对当前记录所作的任何更改或放弃新添加的记录。其语法格式如下：

rs.CancelUpdate

在调用 Update 方法后将无法撤销对当前记录或新记录所作的更改。如果尚未更改当前记录或添加新记录，则调用 CancelUpdate 方法将会产生错误。

5. Find 方法

该方法用于搜索 Recordset 中满足指定标准的记录。如果满足标准，则记录集指针定位在找到的记录上，否则将定位在记录集的末尾。其语法格式如下：

Find (criteria, SkipRows, searchDirection, start)

其中：

- criteria：是一个包含查询条件的字符串，包含指定用于搜索的列名、比较操作符和值的语句。
- SkipRows：可选参数，指定开始查找前需跳过的记录数，默认值为 0。
- searchDirection：可选参数，表示查询方向。有两个取值：adSearchForward 表示向后搜索记录；adSearchBackward 表示向前搜索记录。
- start：可选参数，为指示开始查找位置的书签。

【说明】

（1）criteria“比较操作符”可以是“>”（大于）、“<”（小于）、“=”（等于）、“>=”（大于或等于）、“<=”（小于或等于）、“<>”（不等于）或 like（模式匹配）。criteria 中的值可以是字符串、浮点数或日期。

（2）criteria 中不支持多字段，如"name='abc'"AND "city='sh'" 是不允许的。

5.5　Command 对象

Command 对象是 ADO 中专门用于对数据源执行一组命令和操作的对象，通过该对象可以执行 SQL 语句、数据库中的存储过程等。虽然 Connection 对象的 Execute 方法也能够执行 SQL 语句，但不同的是 Command 对象还提供了参数化查询等专门的运行方式。

一般来说，可以使用 Command 对象的集合、方法、属性进行下列操作。

（1）使用 CommandText 属性定义命令（如 SQL 语句）的可执行文本。

（2）通过 Parameter 对象和 Parameters 集合定义参数化查询或存储过程参数。

（3）可使用 Execute 方法执行命令并在适当的时候返回 Recordset 对象。

创建 Command 对象的语法格式如下：

Set cmd = Server.CreateObject("ADODB.Command")

在后续命令格式介绍中，不妨用 cmd 代表所用的 Command 对象。

5.5.1　Command 对象的常用属性

Command 对象的属性主要有 ActiveConnection、CommandText、CommandTimeout、CommandType 和 State 等。

1．ActiveConnection 属性

该属性定义了 Command 对象所作用的连接。该属性设置或返回 Connection 对象或连接字符串。经常的用法是将一个 Connection 对象赋值给该属性，例如：

```
Cmd.ActiveConnection = conn
```

如果是一个应用连接固定的数据库，也可以将连接字符串存储在某个 Session 对象中，则以后可用如下方式提供到数据源的连接。

```
Cmd.ActiveConnection = Session("database_ConnectionString")
```

2．CommandText 属性

该属性用来指定数据查询信息，指示要对数据库进行的各种操作信息，包括查询、添加、删除、更新等操作。其语法格式如下：

Cmd.CommandText=SQL 语句或数据表名或查询名或存储过程名

通常 SQL 语句使用频率较高。

3．CommandTimeout 属性

该属性用来设置指示等待命令执行的时间（单位为秒），默认值为 30。如果在指定的时间内命令尚未执行完毕，则终止尝试并产生错误。

4．CommandType 属性

该属性用于指定数据查询信息的类型。数据查询信息可以是 SQL 语句或数据表名或查询名或存储过程名，取值可参见 Connection 对象中的介绍。

5.5.2　Command 对象的常用方法

Command 对象的常用方法有 Execute 和 CreateParameter。

1．Execute 方法

该方法用来执行在对象的 CommandText 属性中指定的查询。其用法与 Connection 对象的 Execute 方法基本相同。其语法有两种。

（1）需要返回记录集，使用语句为 Set rs=cmd.Execute()。

（2）无返回记录集的操作，如添加、删除、更新等，使用语句为 cmd.Execute。

2．CreateParameter 方法

该方法用于返回一个 Parameter 对象。其语法格式如下：

Set param = cmd.CreateParameter(name,type,direction,size,value)

其中：

- name 为参数的引用名，在后面引用参数的值时会用到。
- type 指定参数的类型，其值如表 5-14 所示。
- direction 指定参数是输入还是输出，其值如表 5-15 所示。
- size 指定参数的最大长度或最大值。
- value 指定参数的值。

另外，还可以将各个选项分开来写，如下面的两种写法是等价的：

（1）Set param= cmd.CreateParameter(name,type,direction,size,value)

（2）set param= cmd.CreateParameter(name,type,direction,size)

param.value=value

表 5-14　type 值

参　　数	值	说　　明
AddbTimeStamp	135	日期时间类型
AdInteger	3	整数
AdSingle	4	单精度小数
AdDouble	5	双精度小数
AdVarChar	200	变长字符串

表 5-15　direction 值

参　　数	值	说　　明
AdParamInput	1	传入
AdParamOutput	2	传出
AdParamInputOutput	3	传入传出
AdParamReturnValue	4	从子程序返回数据到参数中

5.5.3　Command 对象的数据集合

Command 对象的数据集合有 Parameters 数据集合和 Properties 数据集合，前者表示要传递的命令参数的集合，后者表示 Command 对象的属性集合。Command 对象的 Properties 数据集合与 Connection 对象的使用类似，这里不再叙述。

1．Parameters 参数集合

Parameters 参数集合只有一个 Count 属性，返回该参数集合中 Parameter 参数对象的

数目。

Parameters 集合的常用方法有以下几种。

（1）Append 方法：用于将新建的 Parameter 对象加入到 Parameters 集合中。其语法格式如下：

cmd.Parameters.Append Parameter 对象

（2）Delete 方法：用于删除 Parameter 对象。其语法格式如下：

cmd.Parameters.Delete(index)

其中，index 为删除对象在集合中的索引值，索引值编号从 0 开始，按添加先后顺序排列。

（3）Item 方法：用于取得 Parameters.集合内的某个 Parameter 对象。其语法格式如下：

Set Parameter 对象= cmd.Parameters.item(index)

其中，index 为对象在集合中的索引值。

（4）Refresh 方法：用于重新整理 Parameters 集合。其语法格式如下：

cmd.Parameters.Refresh

2．Parameter 参数对象

Parameter 参数对象代表基于参数化查询或存储过程的 Command 对象相关联的参数。该对象负责传递 Command 对象所需要的 SQL 命令参数，其可以看成是 Command 对象的子对象。

如果使用参数化查询，则可以调用 Parameters 集合中的 Append 方法，并结合 Command 对象中的 CreateParameter 方法，来填充 Command 对象的 Parameters 集合。例如：

```
cmd.CommandType = adCmdText      '常量对应值为 1
cmd.CommandText = "insert into employee(ID, Name) values(?,?)"
set param = cmd.CreateParameter("ID",adInteger,adParamInput,3,4)
cmd.Parameters.Append param
set param = cmd.CreateParameter("NM",adVarChar,adParamInput,255,"mary")
cmd.Parameters.Append param
cmd.Execute
```

【说明】SQL 语句中的问号为数据占位符，Command 对象参数集合中的参数会按照顺序填充相应位置。

5.5.4　通过 Command 对象调用存储过程

在实际开发过程中，Command 对象主要用于建立记录集、执行 SQL 语句或调用存储过程。通过 Command 对象调用存储过程是最常用的方法。使用存储过程可以提高系统的运行速度、减少网络传输时间，同时也为系统提供了一种安全机制。

【例 5-7】　ASP 调用存储过程举例

（1）存储过程

存储过程（getUserName）的功能是访问用户信息表（userinfo），根据用户 ID 得到用户名。其中，UserID 为输入参数，UserName 为输出参数。具体代码如下：

```
CREATE PROCEDURE dbo.getUserName
@UserID int,
@UserName varchar(40) output
As
set nocount on
begin
   if @UserID is null return
   select @UserName=username from dbo.[userinfo] where userid=@UserID
   return
end
go
```

（2）ASP 中调用存储过程

```
-------------------------------------------------------ex5-7.asp-------------------------------------------------------
<%
DIM cmd,UserID,UserName
UserID = 1
Set cmd = Server.CreateObject("ADODB.Command")
cmd.ActiveConnection = Conn             '假设 Conn 是数据库连接对象
cmd.CommandText = "getUserName"         '指定存储过程名
cmd.CommandType = 4                     '表明这是一个存储过程
'以下创建参数
cmd.Parameters.append cmd.CreateParameter("@UserID",3,1,4,UserID)
cmd.Parameters.append cmd.CreateParameter("@UserName",200,2,40)
cmd.Execute
UserName = cmd(1)                       '取得输出参数
%>
```

【注意】

（1）对于存储过程的输入参数，要为 CreateParameter 方法提供 5 个参数，而对于输出参数仅需要为 CreateParameter 方法提供 4 个参数即可，Value 不需要。

（2）参数添加顺序一定要与存储过程中定义的顺序相同，而且各参数的数据类型、长度也要与存储过程中定义的相同。

本 章 小 结

ASP 访问数据库的处理是 Web 应用开发的核心环节。借助本章的技术，可以实现基于浏览器/服务器的数据库应用，用户通过统一的客户浏览器访问分布在 Internet 上的不同数据库应用系统。本章首先简要介绍了 SQL 语句的使用，然后重点介绍了 ADO 的 7 个对象的使用方法，要求读者重点掌握 Connection 对象、Recordset 对象、Command 对象的使用方法；掌握用 ODBC 驱动程序或 OLE DB 链接字符串实现典型数据库连接访问的方法，熟悉记录集的字段访问、记录指针的移动以及翻页访问处理方法；掌握用 Connection 对象、Recordset 对象、Command 对象实现数据库记录的增、删、改的方法；了解用 Command 对

象和 Parameter 对象结合执行存储过程的编程特点。

习　　题

1．选择题

（1）以下 SQL 语句中可查询“用户信息”表中姓“张”的用户的是（　　）。

A．SELECT * FROM 用户信息 WHERE 姓名 LIKE "%张%"

B．SELECT * FROM 用户信息 WHERE 姓名="%张%"

C．SELECT * FROM 用户信息 WHERE 姓名 LIKE "张%"

D．SELECT * FROM 用户信息 WHERE 姓名="张%"

（2）以下 SQL 语句中可查出“用户信息”表中有电子邮件地址（email）的用户的是（　　）。

A．SELECT * FROM 用户信息 WHERE email IS NOT NULL

B．SELECT * FROM 用户信息 WHERE email =NULL

C．SELECT * FROM 用户信息 WHERE email NOT NULL

D．SELECT * FROM 用户信息 WHERE email = NOT NUL

（3）可以实现分组的子句是（　　）。

A．ORDER BY　　B．GROUP BY　　C．HAVING　　D．WHERE

（4）在“教材”表中查出除作者“王小明”和“李晓光”以外其他作者的信息，以下语句中可以实现的是（　　）。

A．SELECT * FROM 教材 WHERE NOT 作者 IN("王小明" , "李晓光")

B．SELECT * FROM 教材 WHERE 作者 NOT IN ("王小明", "李晓光")

C．SELECT * FROM 教材 WHERE 作者 IS NOT "王小明" AND IS NOT "李晓光"

D．SELECT * FROM 教材 WHERE 作者<> ("王小明", "李晓光")

（5）在“教材”表中，将教材数量超过 500 本的教材降价 5%，正确的命令是（　　）。

A．UPDATE 教材 SET 单价=单价*0.95 WHERE 数量>500

B．SELECT 教材 SET 单价=单价*0.95 WHERE 数量>500

C．UPDATE　SET 单价=单价*0.95 WHERE 数量>500 FROM 教材

D．EDIT 教材 SET 单价=单价*0.95 WHERE 数量>500

（6）若要删除“用户信息”表中用户标识为“jsj0911”的记录，正确的命令是（　　）。

A．DROP FROM 用户信息 WHERE 用户标识= "jsj0911"

B．DELETE 用户信息 WHERE 用户标识= "jsj0911"

C．DROP 用户信息 WHERE 用户标识= "jsj0911"

D．DELETE FROM 用户信息 WHERE 用户标识 ="jsj0911"

（7）以下关于 SQL 语句的描述，正确的有（　　）。

A．SQL 命令是区分大小写的

B．利用 UPDATE 命令可同时修改多个字段值

C．每个 SQL 命令只能对一个表进行操作

D．使用 ORDER BY 子句对查询结果进行排序时，默认是升序

（8）在 Connection 对象中，用于存储链接信息的属性是（　　）。

A．ConnectionString　　B．Connection

C．Open　　D．Execute

（9）Connection 对象的 Mode 属性设置为（　　）值时可“以独占方式连接数据源”。

A．0　　B．1　　C．2

D．16　　E．12

（10）在记录集 RS 中可用于返回记录总数的语句是（　　）。

A．num=RS.Count　　B．num=RS.RecordCount

C．num=RS.Fields.Count　　D．num=RS.PageCount

（11）要获得记录集 RS 中当前记录的“产品型号”字段的值，该字段的顺序号为 1，以下用法中不正确的是（　　）。

A．fdvalue=RS(1)　　B．fdvalue=RS.Fields("产品型号")

C．fdvalue=RS("产品型号")　　D．fdvalue=RS.Fields(产品型号).Value

（12）若将记录指针定位到记录集 rs 的最后一条记录，应用（　　）方法来实现。

A．Move　　B．MoveNext

C．MovePrevious　　D．MoveLast

2．问答题

（1）简述 Connection 对象、Command 对象、Recordset 对象的主要功能。

（2）简述 ADO 的各对象与数据集合的关系。

3．完成 SQL 语句的编写

设有 3 个数据表，分别记录学生（STUDENT）、课程（COURSE）、成绩（SCORE）的信息，通过如下 3 条 SQL 语句建表。

```
    CREATE TABLE STUDENT(SNO VARCHAR(3) NOT NULL, SNAME VARCHAR(4) NOT NULL,SSEX VARCHAR(2) NOT NULL, SBIRTHDAY DATETIME, CLASS VARCHAR(5))
    CREATE TABLE COURSE( CNO VARCHAR(5) NOT NULL, CNAME VARCHAR(10) NOT NULL, TNO VARCHAR(10) NOT NULL)
    CREATE TABLE SCORE( SNO VARCHAR(3) NOT NULL, CNO VARCHAR(5) NOT NULL, DEGREE NUMERIC(10, 1) NOT NULL)
```

同时通过 SQL 语句分别插入若干条数据记录，例如：

```
INSERT INTO STUDENT (SNO,SNAME,SSEX,SBIRTHDAY,CLASS) VALUES(108 ,'曾华' ,'男' ,1977-09-01,95033);
INSERT INTO COURSE(CNO,CNAME,TNO) VALUES ('3-105' ,'计算机导论',825)
INSERT INTO SCORE(SNO,CNO,DEGREE) VALUES (103,'3-245',86);
```

试编写实现如下功能的 SQL 语句：

（1）查询 Score 表中成绩在 60～80 分之间的所有记录。

（2）查询 Student 表中 95031 班或性别为“女”的同学的记录。
（3）以 Cno 升序、Degree 降序查询 Score 表中的所有记录。
（4）查询 95031 班的学生人数。
（5）查询 Score 表中最高分的学生学号和课程号。
（6）查询 3-105 号课程的平均分。
（7）查询 Score 表中课程编号以 3 开头的课程的平均分数。
（8）查询 Student 表中不姓“王”的同学的记录。
（9）查询成绩比该课程平均成绩低的同学的成绩表。
（10）查询“计算机导论”课程的所有“男”同学的成绩表。

4．编程题

（1）编写一个班级学生管理程序的应用，提供班级学生的增、删、改、查功能。

（2）编写一个应用实现用户登录检查，数据库表格中存储用户所属角色。角色分为学生、教师、管理员，用户登录后能根据角色转向不同页面。对于不具备相应角色要求的用户，不允许未经登录直接访问页面。

（3）统计用户访问系统的次数，用户每成功登录系统一次，访问次数即增加 1。利用数据库表格字段记录用户的访问次数。

（4）调试以下 ASP 留言簿程序（guestbook.asp），数据库名为 guestbook.mdb，记录留言的数据表 words 有 3 个字段，xm 表示留言者，ly 表示留言内容，sj 表示留言时间。

```
<html>
<body>
<form METHOD="POST" action="guestbook.asp">
<p>姓名:<input TYPE="text" SIZE="20" NAME="xm"></p>
<p>留言:</p>
<p><textarea ROWS="5" COLS="80" NAME="ly"></textarea></p>
<p><input TYPE="submit" VALUE="记入留言簿" NAME="B1">
<input TYPE="reset" VALUE="复原" NAME="B2"></p>
</form>
<hr>
<%
Set conn=Server.CreateObject("ADODB.Connection")
conn.Open "provider=microsoft.jet.oledb.4.0;data source="
     & server.MapPath("guestbook.mdb")
If request("xm")<>"" then
  xm=Request("xm")
  sj = Date()
  sj = sj & " " & Hour(Time()) & ":" & Minute(Time())
  ly=Request("ly")
  sql="INSERT INTO words VALUES('" & xm & "','" & sj & "','" & ly & "')"
  conn.execute sql
End If
sql = "SELECT * FROM words ORDER BY sj DESC"
Set rs = conn.execute(sql)
```

```
Response.Write "<table width=90% ><td>姓名</td><td>发送时间</td><td>留言内容</td>"
do while Not rs.eof %><tr>
<td><%= rs.Fields("xm").Value%></td>
<td><%= rs.Fields("sj").Value%></td>
<td><%= rs.Fields("ly").Value%></td>
<% rs.movenext
loop %>
</table>
</body>
</html>
```

对程序进行调试，并做如下修改。

① 在数据输入域中输入 HTML 标记代码，验证输出。同时将数据库访问的字段输出内容用 Server.HTMLEncode 函数进行变换，进一步验证输出有何变化。

② 增加数据校验代码，当用户输入的姓名和留言为空时进行提示处理。

（5）实现一个多选题的输入管理程序，每道试题包括试题内容、标准答案、难度、所属章节等字段。要求能对各类试题进行增、删、改、查功能。注意，查询支持分页查看。

（6）实现网络教学中的用户管理功能。用户按班级进行管理，能进行用户的增、删、改、查操作。通过下拉列表框选择班级，可看到班级所有学生，并对该班学生进行增、删、改操作。

（7）实现个人记事本，记录每天的重要事情，能对所记内容进行增、删、改、查操作。

第 6 章　JavaScript 脚本语言

JavaScript 是 Netscape 公司推出的一种脚本语言，得到了各种浏览器的支持，利用它可操作和控制浏览器中的各种对象。实际应用中经常利用 JavaScript 编写的客户端脚本完成各类交互功能，如对用户所输入的数据进行有效性检验等。

6.1　JavaScript 的基本语法成分

6.1.1　在网页中插入 JavaScript 代码

在 HTML 网页中插入 JavaScript 语句，应使用 HTML 的<script>标记；或者使用<script language="javascript">，但 language 属性在 W3C 的 HTML 标准中已不再推荐使用。

JavaScript 程序可以放在 HTML 网页的<body>标记或<head>标记中。经常将代码编写为函数形式放在 HTML 的<head></head>中，当用户操作网页的某个对象时触发事件，通过执行事件处理来调用该 JavaScript 函数。

如果某 JavaScript 程序被多个 HTML 网页使用，最好的方法是将这个 JavaScript 程序放到一个后缀名为.js 的文本文件中，在需要使用时，用如下方式引入页面中：

```
<Script Language="JavaScript" src="java/java.js"></script>
```

这样可以提高代码的复用性，减轻代码维护的负担。

【例 6-1】　求一个数的阶乘

```
---------------------------------------------------- ex6-1.htm ----------------------------------------------------------------
<html>
<body>
<script language="javascript">
var n=5;
f=1;
for (k=1;k<=n;k++) {
    f = f*k;
}
alert(n +"!="+f);  //显示 5!值
</script>
<body>
</html>
```

运行程序，将看到页面中弹出一个提示框，如图 6-1 所示。alert 函数的作用是弹出消息框显示提示信息，该函数实际就是后面要介绍的 window 对象的 alert 方法，在使用该方法时可省略前缀 window。



图 6-1　求 n 的阶乘

【说明】

（1）JavaScript 程序由 JavaScript 语句构成，语句间用分号分隔，也可省略分号。用{}括起来的一组语句称为语句块，语句块在语法上相当于一条单独的语句。语句块中的每条语句用分号表示结束，但是语句块本身不用分号。当语句很长时要让代码自动换行，一条语句中不能有硬回车。

（2）为了程序的可读性，可以在程序中为代码写注释。JavaScript 有两种注释，即单行注释和多行注释。单行注释以两个斜杠//开头；多行注释用/*表示开始，用*/表示结束。

（3）程序中的 var 和 for 均为 JavaScript 保留字。保留字是指在 JavaScript 语言中具有特定含义，成为 JavaScript 语法中一部分的那些字。JavaScript 的保留字有 break、delete、function、return、typeof、case、do、if、switch、var、catch、else in、this、void、continue、false、instanceof、throw、while、debugger、finally、new true、with、default、for、null、try。

JavaScript 中还有一些未来保留字，这些字虽然目前没有在 JavaScript 语言中用到，但是将来有可能用到。JavaScript 的未来保留字有 abstract、double、goto、native、static、boolean、enum、implements、package、super byte、export、import、private、synchronized、char、extends、int、protected、throws class、final、interface、public、transient、const、float、long、short、volatile。

6.1.2　数据类型与变量

（1）变量的声明与赋值

变量是用来临时存储数值的容器。在使用一个变量之前，通常要用 var 声明这个变量，同时声明多个变量，变量之间用逗号相隔。例如：

```
var x,y,z;
var a=2, b=5; //声明变量，并为变量赋初始值
```

变量的值是可以变化的。JavaScript 提供了 3 种基本的数据类型，分别为数值型、逻辑型和字符串型，另外还有 undefined 和 null 两个特殊情形。当声明了一个变量而未赋值前，该变量值为 undefined。没有声明的变量为 Null 型，它也只有一个值 null。

另外，JavaScript 中还有引用数据类型，引用类型的变量存储在相应变量单元的内容是一个地址指针，为所指对象在内存单元的位置。对象、数组和函数均属于引用类型。

（2）变量的命名规则

变量名可以是任意长度，它必须符合下列规则：

- 变量名的第一个字符必须是英文字母或下划线符号“_”。
- 变量名的第一个字母不能是数字，而其后的字符可以是英文字母、数字或下划线。
- 变量名不能是 JavaScript 的保留字。

【注意】JavaScript 代码是区分大小写的。例如，变量 myname 和 MyName 表示的是两个不同的变量。同样，对象 window 也不能写成 Window。

（3）变量的作用域

变量的作用域是指定变量的存活范围，在 JavaScript 中，变量的作用域可分为过程级和页面级。所谓过程级指变量只在函数过程内有效，而页面级指变量在整个页面有效。

- 函数中用 var 定义的变量只在函数体内有效，如果出现同名，则屏蔽函数外的变量。
- 如果变量未用 var 定义，则意味着使用函数外同名的变量；如果没有同名的函数外变量，则此变量在函数外仍然有效。

【例 6-2】　变量作用域演示

```
------------------------------------------------------ex6-2.htm------------------------------------------------------
<script language="javascript">
var x=13,y=29;                          //页面级变量
  function test() {
  var num,y=10;                         //过程级变量，这里 y 隐藏外部定义的 y
  num=x+y;
  x++;
alert("内部的 num 的值为："+num);       //显示 num 为 23
}
test();                                 //调用 test 函数
alert("x 的值为："+x);                  //显示 x 为 14
alert("外部的 num 的值为："+num);       //无显示，这里不能访问 num
</script>
```

【说明】本例编写了一个函数 test()，并在页面代码中调用该函数，运行程序可看到两个提示框，一个是在函数内显示 num 的值，另一个是在外部显示 x 的值，但无法看到外部显示 num 的值的提示框。因为 num 为过程级变量，在函数外不能访问，并因此出现访问异常，代码将停止执行。

6.1.3　JavaScript 运算符

（1）算术运算符

算术运算符有+（加法）、-（减法）、*（乘法）、/（除法）、%（求余数）、++（递增）、--（递减）。

其中，运算符（+）既可用于数值运算，又可用于字符串的拼接。

（2）关系运算符

关系运算符有>（大于）、>=（大于等于）、<（小于）、<=（小于等于）、!=（不等）、==（等于）、===（全等于）、!==（不全等于）。

其中，===（全等于）表示不仅值相等，而且数据类型也相等。例如，如果有 x=2;y="2";，则 x===y 为 False 值，而 x==y 为 True。!==（不全等于）表示不等于或类型不同。

（3）逻辑运算符

逻辑运算符有与（&&）、或（||）、非（！）。

（4）位运算符

位运算是对操作数以二进制比特（bit）位为单位进行的操作运算，其操作数和结果都是整型量。下面列出几种位运算符和相应的运算规则，如表 6-1 所示。

表 6-1　位运算符

运　算　符	用　　法	操　　作
~	~op	结果是 op 按比特位求反
>>	op1 >> op2	将 op1 右移 op2 个位（带符号）
<<	op1 << op2	将 op1 左移 op2 个位（带符号）
>>>	op1 >>> op2	将 op1 右移 op2 个位（不带符号的右移）
&	op1 & op2	op1 和 op2 都是 True
|	op1 | op2	op1 或 op2 是 True
^	op1 ^ op2	op1 和 op2 是不同值

（5）赋值运算符

赋值组合运算符是指在赋值运算符的左边有一个其他运算符，例如：

```
x+=2;   //相当于 x=x+2
```

其功能是先将左边变量与右边的表达式进行某种运算，再把运算的结果赋给变量。能与赋值运算符结合的运算符包括算术运算符（+、−、*、/、%）和位运算符（&、|、^、>>、<<、>>>）。

6.1.4　内置函数

JavaScript 中有如下几种常用的全局函数。

- eval(string)：返回字符串表达式中的值，如 eval("2+4*5")的结果为 22。
- isFinite(exp)：确定一个变量是否有界，如果有界则返回 True，否则为 False。
- isNaN(n)：确定一个变量是否为 NaN，如果是则返回 True，否则返回 False。
- parseFloat(floatstring)：返回实数，若字符串不是以数字开头，则返回 NaN。
- parseInt(numberstring,radix)：将第一个参数所给的数字串转换为 radix 指定的进制的数值。若字符串不是以数字开头，则返回 NaN。如果第二个参数为默认参数，则返回十进制数。
- escape(string)：对字符串的特殊字符进行转换编码，以满足特殊处理的要求。
- unescape(string)：完成 escape()函数的逆操作，将编码后的串恢复到源串。

【例 6-3】　escape 和 unescape 函数的演示

```
-----------------------------------------------------ex6-3.htm-----------------------------------------------------
<script type="text/javascript">
var test1="welcome to ecjtu!"
test1=escape(test1)                          //编码
document.write (test1 + "<br />")
```

```
test1=unescape(test1)                    //反编码
document.write(test1 + "<br />")
</script>
```

运行结果如下：

welcome%20to%20ecjtu%21

welcome to ecjtu!

【**说明**】这里利用了 document 对象的 write 方法向页面文档中写入要输出显示的信息。

6.2　程序流程控制语句

6.2.1　条件语句

JavaScript 条件语句有 if 语句和 switch 语句两种。

（1）if 语句

if 语句语法格式如下：

if (条件表达式) statement1　[else　statement2]

其中，else 部分可没有；如果 statement1 和 statement2 由多条语句组成，则必须用大括号（{ }）括起来。

（2）switch 语句

switch 语句用于多分支处理，其语法格式如下：

```
switch (expression)
 {
   case label1 :
       statement1
       break
   case label2 :
       statement2
       break
   ...
   default :
       statementdefault
 }
```

如果 expression 的值为 label1，则执行 statement1 代码；如果 expression 的值为 label2，则执行 statement2 代码；以此类推。如果 expression 不符合任何情形，则执行 default 内的 statementdefault 代码。switch 条件语句中的 break 表示 switch 语句结束。如果没有使用 break 语句，则找到匹配的 label 开始执行，后面的多个 label 块都将被执行，直到遇到 break 为止。

【例 6-4】　今天是上课还是休息

```
--------------------------------------------------------ex6-4.htm--------------------------------------------------------
<script type="text/javascript">
var d = new Date()
theDay = d.getDay()
switch (theDay){
  case 6:
  case 0:
    alert("休息日")
    break
  default:
    alert("上课日")
}
</script>
```

6.2.2　循环语句

JavaScript 循环语句有 for 循环语句、do…while 循环语句、while 循环语句。

（1）for 循环语句

for 循环语句的语法格式如下：

for (exp1;exp2;exp3) { statement }

其中，exp1 为循环前的初值设置；exp2 为条件表达式，只有当 exp2 为 True 时才执行循环体；exp3 是一个更新循环控制变量的表达式，它使循环离结束更近。

（2）while 循环语句

while 循环语句的语法格式如下：

while(条件){statement}

当条件为 True 时，反复执行循环体，要注意 statement 中必须有让循环条件能发生变化的操作，从而避免出现死循环。

（3）do…while 循环语句

do…while 循环与 while 循环相似，不同之处在于它总是至少运行一次，因为其是在循环的末尾检查条件，而不是在开头。

【例 6-5】　求一个数的各位数字之和

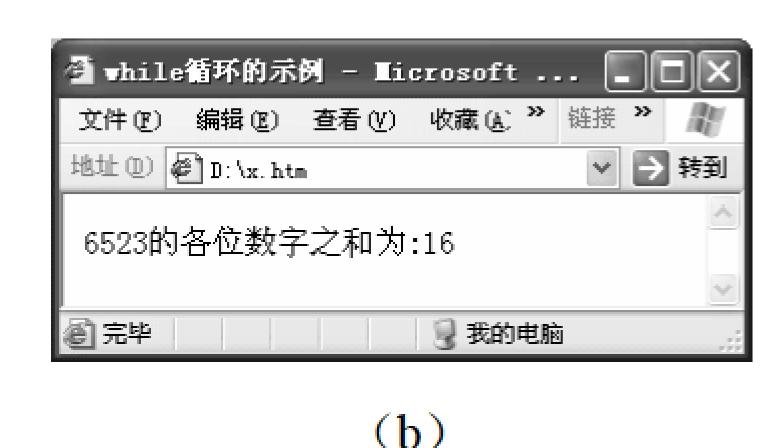
```
--------------------------------------------------------ex7-5.htm--------------------------------------------------------
<script type="text/javascript">
 var x = window.prompt("请输入一个整数","");          //读一个输入串
 var y = parseInt(x);                                 //将字符串转化为整数
 var n=0;
 while (y>0) {
    n=n+y%10;                                         //取 y 的最低位数字
    y=Math.floor(y/10);                               //取出结果的整数部分，y 去掉最低位
  }
```

```
  document.write(x+"的各位数字之和为："+n);
</script>
```

执行代码，首先执行 window 对象的 prompt 方法，弹出“Explorer 用户提示”对话框，提示等待用户输入数据，如图 6-2（a）所示。在输入数据后，单击“确定”按钮，即可在网页文档中输出结果，如图 6-2（b）所示。



（a）



（b）

图 6-2　求一个数的各位数字之和

（4）使用 break 和 continue 语句

break 和 continue 语句均可用于循环内，执行 break 语句将中断一个循环的运行。执行 continue 语句则越过余下的代码块而直接跳到循环的下一次重复中。

【例 6-6】　输出 1～10 之间的奇数

```
---------------------------------------------------ex7-6.htm---------------------------------------------------
<script type="text/javascript">
var x ;
for (x=1; x<=10; x++) {
    if (x%2==0)
        continue; //如果 x 被 2 整除，则跳过后面的代码，开始下一次重复
    document.write (x + "<br>");
}
</script>
```

6.3　内 置 对 象

JavaScript 语言是基于对象的，程序中可以根据需要创建对象，每一个对象均有自己的属性和方法。对象分为内置对象、浏览器对象、文档对象、用户自定义对象 4 种不同类型的对象。

常用的内置对象有 String 对象、Math 对象、Array 对象和 Date 对象。内置对象通过使用 new 关键字和对象构造函数创建对象实例，并可以通过“.”运算符来访问对象实例的属性和方法。

6.3.1　String 对象

创建一个 String 对象，可用如下语法：

var 变量名 = new String("字符串");

事实上，任何一个字符串常量都是一个串对象，例如：

```
var msg="Welcome";
```

以上两种形式定义的字符串在使用上没有差异，两者的区别是 typeof 的返回值不同，后者是 string，前者是 object。

（1）属性

length：字符串的长度，即字符串中字符的个数。

（2）方法

- ❑ charAt(index)：返回字符串中 index 处的字符。
- ❑ indexOf(substr,[fromIndex])：返回字符串中 substr 子串第一次出现的位置，如果未找到，则返回-1；如果给定了 fromIndex，则从字符串内该位置开始搜索。
- ❑ lastIndexOf(substr,[fromIndex])：从字符串的尾部向前搜索 substr，并返回第一个出现的位置，如果未找到，则返回-1。
- ❑ substring(indexA,indexB)：从字符串获取自 indexA 到 indexB 的子串。
- ❑ toLowerCase()：返回的字符串是将字符串中所有字符全部转换成小写。
- ❑ toUpperCase()：返回的字符串是将字符串中所有字符全部转换成大写。

可为字符串增加相应的 HTML 标记的方法有 big()、blink()、bold()、fixed()、italics()、small()、sub()、strike()、sup()、fontColor(color)、fontSize(size)。

【例 6-7】　用字符串的处理方法求一个数字串的各位数字之和

```
---------------------------------------------------------ex6-7.htm---------------------------------------------------------
<script type="text/javascript">
var x = window.prompt("请输入一个整数","");    //读一个输入串
var k=0;
var n=0;
 do  {
     n = n+ (x.charAt(k)-'0');                        //取 x 的第 k 个位置的数字
     k++;
  } while (k<x.length)
  document.write(x+"的各位数字之和为："+n);
</script>
```

【说明】从字符串中逐个取出各位数字字符，注意，将数字字符转化为数字值的一个简单方法是将字符与代表零的字符进行相减，这是利用数字字符的编码顺序递增 1 的特点。

【例 6-8】　在状态栏实现滚动字幕

```
---------------------------------------------------------ex6-8.htm---------------------------------------------------------
<script language="javascript">
var msg = "欢迎光临访问我的网站!";
var i=1;
function scroll()  {
    mess = msg.substring(i,msg.length)+""+msg.substring(0,i);
    window.status=mess;
    i++;
    if (i>=msg.length) i=1;                   //滚动到消息末尾，从头开始
    setTimeout("scroll()",200);               //延时决定滚动速度
```

```
}
</script>
<body onload="scroll()"></body>
```

【说明】setTimeout 为 window 对象的一个方法，用来设置多少毫秒后调用指定函数。本例调用 scroll()函数引入了事件机制。通过 body 标记的 onload 属性设置定义页面装载事件触发执行 scroll 函数。

6.3.2　Array 对象

1．创建数组对象实例

创建数组对象实例有如下几种形式：

格式 1：var x= new Array();　　　　　//数组的初始大小不定

格式 2：var y= new Array(size)　　　　//创建初始大小为 size 的数组

格式 3：var z= new Array(1,2,3,5,6,8)　　//按提供的初始化数据创建数组

格式 1 创建的数组是动态数组，也即数组的大小是不定的；格式 2 产生一个指定大小的数组，但数组的大小在程序运行时仍然可变；格式 3 按括号中提供的初始化数据创建数组，数组的大小取决于初始化数据的个数。

2．Array 对象的属性和方法

Array 对象的 length 属性表示一个数组中元素的个数。

Array 对象的常用方法有以下几种。

- ❑ reverse()：将一个数组按倒序重排。
- ❑ sort()：按由小到大的顺序对数组元素重新排序。

3．数组元素的访问

数组元素的访问格式如下：

数组名[下标值]

例如：

```
x[1]=12   //给下标为 1 的元素赋值为 12
```

【注意】

（1）同一数组的元素赋值类型可以不同。例如：

```
x[0]= 20;            //数值型
x[1]= "你好";        //字符串型
x[2]= true;          //逻辑型
```

（2）数组元素还可以是对象，当数组元素为一个数组时，就可得到一个二维数组。例如：

```
var arr=new Array(5);
for (k=0;k<arr.length;k++)
  arr[k]=new Array(4);
```

这样，就创建了一个 5 行 4 列的二维数组。

二维数组的元素引用如下：

数组名[第一维下标][第二维下标]

例如，arr[2][3]代表引用第 3 行第 4 列的元素。

（3）数组的大小会随元素赋值访问动态变化。例如，运行以下代码，结果为“数组大小=8”。

```
<script type="text/javascript">
  var d = new Array(5);
  d[7]=123;
  alert("数组大小="+d.length);
</script>
```

6.3.3　Date 对象

Date 对象封装了有关日期和时间的操作，它没有任何属性，但有很多方法。创建日期对象的语法格式如下：

Now = new Date([参数表]);

最常用的是无参情形，new Date()将以系统当前的日期、时间产生一个对象实例。

常用方法如下。

- getDate()：当月日期部分，范围为 1～31。
- getDay()：返回该星期的第几天，范围为 0～6。
- getMonth()：返回月份，范围为 0～11。
- getTime()：返回自 1970 年 1 月 1 日 00:00:00 以来的毫秒数。
- getYear()：在 2000 年以前返回后两位；在 2000 年以后返回 4 位。
- getHours()：返回小时数。
- getMinutes()：返回分钟数。
- getSeconds()：返回秒数。
- setYear(int)：设置年。
- setDate(int)：设置当月号数。
- setMonth(int)：设置当月份数。
- setHours(int)：设置小时数。
- setMinutes(int)：设置分钟数。
- setSeconds(int)：设置秒数。
- setTime(int)：设置毫秒数。

6.3.4　Math 对象

Math 对象包含用来进行数学计算的属性和方法，其属性就是一些标准的数学常量，其

方法构成了数学函数库。

【注意】Math 对象的属性和方法均直接用 Math 作为前缀使用，另外，不能用 new 运算符创建 Math 对象。

1．常用属性

Math 对象的常用属性如表 6-2 所示。

表 6-2　Math 对象的常用属性

常　　量	数 学 含 义	近　似　值
Math.E	e	2.71828459045
Math.LN2	ln2	0.69314718056
Math.LN10	ln10	2.30258509299
Math.LOG2E	log2e	1.44269504889
Math.LOG10E	log10e	0.43429448190
Math.PI	圆周率	3.14159265359

2．常用方法

Math 对象的常用方法如下。

- Math.floor(n)：返回小于等于 n 的最大整数。
- Math.round(n)：返回最靠近 n 的整数。
- Math.max(n1,n2)：取 n1、n2 中的最大值。
- Math.min(n1,n2)：取 n1、n2 中的最小值。
- Math.pow(n1,n2)：返回 n1 的 n2 次方。
- Math.random()：产生 0～1 之间的随机数。
- Math.sqrt(n)：返回 n 的平方根。
- Math.abs(n)：返回 n 的绝对值。
- Math.exp(n)：返回 e 的 n 次方。
- Math.log(n)：返回 n 的自然对数。
- Math.sin(n)：返回 n 的正弦值。

6.4　自定义函数

6.4.1　函数的定义

语法格式如下：

```
function　函数名(形式参数){
   函数体
   return　表达式
}
```

函数的定义通常放在网页的<head></head>标记部分。

【注意】如果函数调用时参数为引用型实参，那么在函数内修改实参变量值就会改变它所指向的原始数据。

6.4.2　函数的调用

格式 1：var name = 函数名(实际参数)　　//有返回值时使用
格式 2：函数名(参数值)　　//无返回值时使用
JavaScript 的函数采用传值方式进行传递，也就是将实参单元的值传递给形参单元。

【例 6-9】　控制一行中显示字符的最大长度

```
--------------------------------------------------------ex6-9.js--------------------------------------------------------
<script type="text/javascript">
function convert(str,n) {          //字符串 str 在显示时最长限制为 n 个英文字符
var result="";
var count=0;
var i,c;
var len=str.length;
i=0;
while (i<len) {
    c= str.charAt(i);              //取串的第 i 个位置的字符
    if (c.charCodeAt(0)<299)       //根据字符编码判断是否为汉字
        count+=1;                  //英文字符
    else
        count+=2;                  //中文字符
    result = result+c;
    if (count>=n)   {
        result+="…";               //如果超出长度 n，则用省略号代替
        break;
    }
  i=i+1;
}
return result;
}
</script>
```

【说明】此函数在显示网页动态内容时有用，它可以实现列表项内容的整齐排列，否则会造成有的行内容可能很长，从而影响整个页面的显示排列效果。网络教学系统中导航菜单的显示就利用了该函数进行显示处理，另外还有一些网站新闻标题的显示也可能会用到。

6.5　用户自定义对象

用户可以自定义对象，为对象定义属性和方法。JavaScript 的对象是无类别的，它对属性的名字和值没有约束。另外，JavaScript 对象的属性和方法还可以动态增、删。

6.5.1　自定义对象创建方式

1．对象字面量（也称对象初始化器）

在 JavaScript 中，对象是“名/值”对的集合。对象字面量是一个名值对列表，每个名值对之间用逗号分隔，并用一个大括号括起。定义格式如下：

```
objectName = {
   property1:value1,
   property2:value2,
   …,
   propertyN:valueN
}
```

其中，各名值对表示对象的一个属性，名和值之间用一个冒号分隔。property 代表对象的属性名，value 则是对象的属性值，值可以是字符串、数字或对象三者之一。

例如：

```
<script type="text/javascript">
var user = {
      name: "Smith",
      age: "24"
 }
alert(user.age);
</script>
```

2．构造函数方式

通过 function 定义编写一个构造函数，并通过 new 方式来创建对象，例如：

```
function User(name,age){
   this.name=name;
   this.age=age;
}
var oneuser = new User();
```

其中，this 是对当前对象的引用。

值得一提的是，JavaScript 对象的属性和方法是动态可扩展的。对于已经实例化的对象 obj，还可以动态增加和删除它的属性与方法。

（1）动态增加对象属性

```
obj.newPropertyName=value;
```

（2）动态增加对象方法

形式 1：

```
obj.newMethodName = method     //method 为一个对象外已经存在的方法
```

形式 2：

```
obj.newMethodName = function(arg1,…,argN){ }
```

（3）动态删除对象属性

```
delete obj.propertyName
```

（4）动态删除对象方法

```
delete obj.methodName
```

6.5.2　JavaScript 对象的操作

JavaScript 提供了几个用于操作对象的语句、关键字及运算符。

1．for…in 语句

JavaScript 提供了一种特别的循环方式来遍历一个对象的所有属性。其语法格式如下：

For(对象属性名称 in 已知对象名称)

【例 6-10】　遍历对象的所有属性

```
------------------------------------------------------ex6-10.htm------------------------------------------------------
<script type="text/javascript">
var myObject = new Object();
myObject.sitename = "DZF 学练园地";        //动态增加属性
myObject.siteurl = "cai.ecjtu.jx.cn";
for (prop in myObject)
{
    document.write("属性 '" + prop + "' 为 " + myObject[prop]);
    document.write("<br>");
}
</script>
```

【说明】for…in 语句自动地获取对象的每个属性名为 prop 变量赋值，而 myObject[prop]则获取对应的属性值。

2．with 语句

其语法格式如下：

with (object) { …… }

所有在 with 语句后大括号中的语句都是在后面 object 对象作用域中的。下面的代码段显示两种访问 HTML 表单的元素的方法。方法 1 是普通的方法，方法 2 是使用 with 语句的方法。

方法 1：

```
frames[1].document.forms[0].address.value
```

方法 2：

```
with (frames[1].document.forms[0]) {
        address.value = "";
        name.value = "";
}
```

一般情况下，方法 2 用在需要批量访问一个对象的属性或方法的地方。

6.5.3　定义对象属性

在 JavaScript 中可以为对象定义 3 种类型的属性，即私有属性、实例属性和类属性，与 Java 类似，私有属性只能在对象内部使用，实例属性必须通过对象的实例进行引用，而类属性通过类名进行引用。

1．定义私有属性

私有属性只能在构造函数内部定义与使用。

语法格式：var propertyName=value;

例如：

```
function User(age){
    this.age = age;
   var isChild = age<12;                  //isChild 为私有属性
   this.isLittleChild = isChild;
}
var user = new User(15);
alert(user.isLittleChild);                //正确的方式
alert(user.isChild);                      //报错，对象不支持此属性
```

2．定义实例属性

定义实例属性存在 3 种方式。

（1）原型（prototype）方式

在 JavaScript 中，函数就是对象，每个对象实例包括一个指向该原型属性和方法的 prototype 属性，可以用它为原型的对象指定属性和方法。

语法格式：functionName.prototype.propertyName = value

例如：

```
function User() {   }
User.prototype.name = "张三";
User.prototype.age = 18;
var user = new User();
alert(user.age);
```

（2）this 方式

语法格式：this.propertyName = value

例如：

```
function User(name,age) {
   this.name = name;
   this.age =   age;
}
user1 = new User("李四",25)
alert(user1.age);
```

（3）下标（index）方式

语法格式：obj[index] = value

例如：

```
function User(name) {
   this.name = name;            //this 方式定义
   this.age = 18;
   this[1] = "ok";              //index 方式定义
}
var user2 = new User("王五");
alert(user2.name);              //正常方式引用
alert(user2[1]);                //index 方式引用
```

【注意】通过 index 方式定义的必须使用 index 方式来引用，而没有通过 index 方式定义的则必须以正常方式引用。

3．定义类属性

语法格式：functionName.propertyName = value

例如：

```
function User(){ }
User.MAX_AGE = 200;             //类属性定义
User.MIN_AGE = 0;
alert(User.MAX_AGE);
```

6.5.4　定义对象方法

在 JavaScript 中可以为对象定义 3 种类型的方法，即私有方法、实例方法和类方法。与 Java 类似，私有方法只能在对象内部使用；实例方法必须在对象实例化后才能使用；类方法必须通过类名使用。

1．定义私有方法

私有方法必须在构造函数体内定义，而且只能在构造函数体内使用。

语法格式：function methodName(arg1,…,argN){ }

例如：

```
function User(name){
   this.name=name;
```

```
  function getNameLength(nameStr){              //私有方法
    return nameStr.length;
  }
  this.nameLength = getNameLength(this.name);
}
```

2. 定义实例方法

定义实例方法可以有多种方式。

（1）this 方式。在构造函数内部使用，语法格式有以下两种。

形式 1：this.methodName = method;

形式 2：this.methodName = function(arg1,…,argN){ };

在以上语法描述中，method 代表外部的一个方法，methodName 是要定义的对象的方法，形式 1 的含义是将外部的一个方法直接赋给对象的某个方法。

【注意】在 JavaScript 等式右边直接写方法名表示引用一个已定义的方法。

（2）prototype 方式。在构造函数外部使用，语法格式以下两种。

形式 1：functionName.prototype.methodName = method;

形式 2：functionName.prototype.methodName = function(arg1,…,argN){ };

【例 6-11】　对象方法的定义形式

```
------------------------------------------------------ex6-11.htm------------------------------------------------------
<script language="javascript">
function User(name){
    this.name=name;
    this.getName = function(){              //this 方式的形式 2
                return name;
            };
}
User.prototype.show = say;                  //prototype 方式的形式
function say(someobj) {                     //显示对象 someobj 的所有成员
   for (k in someobj)
      document.write(k+"<br>")
}
obj = new User("张三");
document.write(obj.getName()+"<br>");
obj.show(obj);
</script>
```

在浏览器中查看文件，页面显示结果如下：

张三

name

getName

show

【注意】方法的内外放置不是任意的，需根据方法要访问的数据决定。上面的 getName 方法不能采用引用外部方法的形式，否则将无法访问对象的 name 属性。

（3）对象字面量形式。对象字面量提供了一种非常方便地定义对象属性及方法的表示形式。

```
<script type="text/javascript">
var user = {
  name: "Smith",                    //定义 name 属性
  getName:function() {              //定义 getName 方法
       return this.name;
  },
  setName:function(name) {          //定义 setName 方法
       this.name=name;
  }
}
user.setName("Mary");               //使用 setName 方法
alert(user.getName());              //使用 getName 方法
</script>
```

【说明】在对象字面量定义时，名值之间用冒号分隔，各数据项或方法之间用逗号分隔。

3．定义类方法

类方法需要在构造函数外部定义，且以构造函数名作为前缀。语法格式有以下两种。

形式 1：functionName.methodName = method;

形式 2：functionName.methodName = function(arg1,⋯,argN){ };

【注意】类属性与方法可以在任何地方通过类名来使用，但不能通过对象的实例来引用。这一点与 Java 不同，在 Java 中静态成员可以通过实例来访问。

最后要说明的是，JavaScript 的一个重要应用是根据事件驱动执行脚本代码，进而实现页面显示更改。该部分内容将在第 8 章中进行详细介绍。

本 章 小 结

JavaScript 是客户端编程广泛使用的脚本语言，本章主要介绍了其基本语法成分、内置对象的使用，还有自定义函数与自定义对象的操作访问方法。JavaScript 的变量类型取决于赋值的数据类型。JavaScript 是一种基于对象的语言，其定义对象的形式与 Java 有较大的不同，定义对象的形式有多种，而且对象的属性和方法可以动态增加和删除。在第 8 章和第 10 章中还将介绍 JavaScript 的高级应用。

习　　题

1．选择题

（1）JavaScript 的数据类型主要有（　　）。

A．字符串值　　B．整数　　C．浮点数
D．逻辑值　　E．布尔值

（2）关于变量的命名规则，下列说法正确的是（　　）。
A．首字符必须是大写、小写的字母、下划线（_）或美元符号（$）
B．后续的字符可以是字母、数字、下划线或美元符号
C．变量名称不能是保留字
D．长度是任意的
E．区分大小写

（3）有关 JavaScript 语句，下列说法正确的是（　　）。
A．单行注释语句是在需要注释的行前面加//
B．多行注释语句是在需要注释的文字两端加/* 注释文字*/
C．with 语句的功能是为一段程序建立默认对象
D．JavaScript 中没有 if…else 语句
E．JavaScript 中只有 while 语句，而没有 do…while 语句

（4）以下生成对象的方法中，正确的是（　　）。
A．var z = new Boolean(a);
B．var str = "JavaScript";
C．fruit=new Array(3);
D．today1=new Date(2008,10,1);
E．today=new Date("October 1,2008");

（5）求一个表达式的值，可以使用的函数有（　　）。
A．eval()　　B．isNaN()
C．parseInt()　　D．parseFloat()

（6）关于 JavaScript 函数，下列说法正确的是（　　）。
A．函数是独立于主程序的、具有特定功能的一段程序代码块
B．函数可以不用 function 关键字
C．函数的命名规则同变量的命名规则一样
D．函数必须使用 return 语句
E．函数在调用时直接用函数名，并为形式参数赋值

（7）在 JavaScript 中，用于产生输出的是（　　）。
A．document.write()　　B．window.write()
C．document.confirm()　　D．write "The Undefined Function"

（8）下列（　　）是获得某时间对象的月份的方法 getMonth()的特点。
A．获取的月份是从 1 开始计数的
B．获取的月份是从 0 开始计数的
C．获取的月份是英文单词的缩写，如 6 月份为 Jun
D．无法获取月份

2. 设计题

（1）根据时间段的不同，在网页中显示不同的问候语，若时间在 12 点以前，则输出“早上好！”的问候语，颜色为绿色；若时间在 12～18 点，则输出“下午好！”的问候语，颜色为黄色；若时间在 18 点以后则输出“晚上好！”的问候语，颜色为黑色。

（2）编写一个 JavaScript 程序，在页面上通过输入对话框出 10 道两位数的加法练习题，若用户回答正确，则加 10 分，最后用对话框显示用户得分。

（3）以下程序在页面上动态显示当前时间，将程序补充完整。

```
<html>
<head>
<title>动态显示日期时间</title>
<script type="text/javascript">
    var days = new ___①___("日","一","二","三","四","五","六");
    function show(){
        var ___②___=new Date();
        var y,m,date,day,hs,ms,ss,theDateStr;
        y= today.___③___;                      //4 位整数表示的年份
        m= today.getMonth();                   //月
        date= today.___④___ ;                  //日
        day= today.getDay();                   //星期
        hs= today.getHours();                  //时
        ___⑤___ = today.getMinutes();          //分
        ss= today.getSeconds();//秒
        x.innerHTML = y+"年"+___⑥___+"月"+date+"日  星期"+days[day]+" "+hs+":"+ms+":"+ss;
        window.setTimeout(___⑦___,1000);
    }
___⑧___
</head>
<body onLoad="show()">
    <div   id="x"></div>
</body>
</html>
```

（4）利用 JavaScript 实现一个人与计算机对拿火柴的游戏。火柴数量用随机函数产生，范围是 50～100 根之间，每次限拿数量为 3 根，谁先拿也由随机数决定，拿到最后一根火柴的为胜者。利用 prompt 函数提示用户输入；利用 alert 函数显示结果。

第 7 章　层叠样式表 CSS

CSS 的全名是 Cascading Style Sheets，即“层叠样式表”，简称样式表，用来定义网页的显示样式。在 HTML 中，网页内容和网页样式是混为一体的，如果想修改某个网页的样式，就需要对该网页进行逐句修改，非常麻烦。单纯使用 HTML 的标记和属性，有时达不到需要的效果。层叠样式表是一系列格式设置规则，使用 CSS 可以灵活控制页面外观，包括从精确的布局定位到特定的字体和样式。它提供了便利的更新功能，当更新某 CSS 样式时，使用该样式的所有文档的格式都会自动更新为新样式。

7.1　样式表的定义与引用

CSS 格式设置规则由选择器和声明两部分组成。选择器是标识格式元素的术语，HTML 中的所有标记都可以作为选择器，如 p、td 等，另外，选择器还可以用 class 和 id 选择符来表示。声明用于定义元素样式，它由属性和值两部分组成。CSS 语法的一般格式如下：

选择符{ 属性: 属性值; 属性: 属性值; …}

例如：

```
h1{ color:red;font-family:宋体}
```

1．标记选择符

标记选择符是指以 HTML 标记作为名称的选择符，如 body、p、tr、h1 等。通过 CSS 可以重新定义这些标记的显示样式，如 p 选择符就可以定义页面中所有<p>标记的样式风格。

对于标记选择符定义的样式，只要在网页中有这些标记，这些标记就会自动按照 CSS 样式的风格显示。

例如，以下代码段定义了<h1>标记的字体大小和字体颜色。

```
<style type="text/css">
    h1{
        font-size: 18px;
        color: #F600FF;
    }
</style>
```

【注意】在使用标记选择符时，如果标记的某些属性在 CSS 中没有定义，则标记就会继承原有的属性，就如本例中的 h1、h2、h3 标记中没有定义字体为粗体，但是字体显示的效果却为粗体。

2．class 选择符

使用类选择符定义样式有以下两种形式。

形式 1：标记名.类名{规则 1；规则 2；…}

形式 2：.类名{规则 1；规则 2；…}

第一种形式定义的样式只能应用于指定名称的标记上，例如：

```
p.back{background-color:#888888;}
```

只能在 p 标记上应用样式，为标记加入 class 属性。例如：

```
<p class="back"> 这段文字的背景色应用了样式</p>
```

第二种形式定义的样式则不受限制，可应用于所有标记。例如：

```
.red{ font-size:12px;color:red;font-family:宋体}
```

同样可以为 p 标记应用该样式。例如：

```
<p class="red"> 使用 class 选择符</p>
```

【应用技巧】如果使用了标记选择符，则可以使用类选择符去改变个别已定义了样式的标记选择符的样式。

3．id 选择符

id 选择符用于定义一个元素独有的样式，它与类选择符的区别在于，id 选择符在一个 HTML 文件中只能使用一次，而类选择符可以多次引用。id 选择符定义的格式如下：

#id 名{规则 1；规则 2；…}

例如：

```
#red{font-size:12px;color:red;font-family:宋体}
```

要为某个标记应用该样式，需要在标记中加入 id 属性，属性值为指定的 id 名。例如：

```
<p id="red">应用 id 选择符</p>
```

4．上下文选择符

上下文选择符定义嵌套标记的样式，标记中间加空格分开。例如：

```
td p {font-size:12px;color:red;font-family:宋体}
```

表示在 HTML 文档中出现嵌套标记<td><p>…</p></td>的地方引用该样式。例如：

```
<table><tr><td><p>嵌套标记的样式使用</p></td></tr></table>
```

5．群定义

多个选择符要定义同样的样式，可以用逗号分隔。例如：

```
p,h1,a {font-size:12px;color:red;font-family:宋体}
```

群组选择符可以简化 CSS 的编写，使用群组选择符可以定义多个相同样式的标记选择符。本例中的 h1、p、a 具有相同的样式，如果想使网页中的几个 p 标记显示出其他样式，可以采用上下文选择符的方法。

6．伪类

伪类是特殊的类，它可以被支持 CSS 的浏览器自动识别，根据标记的状态自动应用样式。需要注意的是，伪类不能用 class 属性来指定。伪类定义的格式如下：

选择符:伪类{规则 1；规则 2；…}

伪类最常见的应用是超链接（a 标记），超链接有各种访问状态，包括已访问链接（visited link）、可激活链接（active link）等。例如：

```
a:link {color: #FF0000}       /* 未被访问的链接 红色 */
a:visited {color: #00FF00}  /* 已被访问过的链接 绿色 */
a:hover {color: #FFCC00}    /* 鼠标悬浮在上的链接 橙色 */
a:active {color: #0000FF}   /* 鼠标点中激活的链接 蓝色 */
```

【注意】由于 CSS 优先级的关系（后面比前面的优先级高），在写 a 的 CSS 时，一定要按照 a:link、a:visited、a:hover、a:actived 的顺序书写。

7.2　样式表的种类

CSS 按其位置可以分为内嵌样式、内部样式表和外部样式表 3 种。

1．内嵌样式

内嵌样式是写在 HTML 标记中的，只对所在的标记有效。其语法格式如下：

<标记 style="属性: 属性值; 属性: 属性值; …">…</标记>

例如：

```
<P style="font-size:20pt; color:red">…</p>
```

该样式定义<p></p>中的文字是 20pt 字体，字体颜色是红色。

【应用技巧】一般情况下，当浏览器显示表格中的内容时，各单元格显示的实际宽度依赖表格内容变化，而在动态网页中，数据信息的长度也是变化的。为了让表格显示时有一个固定宽度，同时让内容在显示超宽时自动转行，可进行如下设置：

首先在表格标记中设置样式 style="table-layout:fixed"，然后在某单元格标记中设置样式 style='word-wrap:break-word'。那么，这个单元格就不会因内容太多而延长表格宽度，从而保证了页面显示效果的整齐。

2．内部样式表

内部样式表在 HTML 的<head></head>内定义，只对所在的网页有效。在 FrontPage 中创建内部样式，选择“格式”→“样式”命令，可弹出如图 7-1 所示的对话框，在该对话框中可新建和修改样式。

【例 7-1】　内部样式表的定义与使用

```
---------------------------------------------------------ex7-1.htm---------------------------------------------------------
<HTML>
<HEAD>
<STYLE type="text/css">
    H1.mylayout {border-width:1; border:solid; text-align:center; color:red}
</STYLE>
</HEAD>
<BODY>
<H1 class="mylayout"> 这个标题使用了样式。</H1>
<H1>这个标题没有使用样式。</H1>
</BODY>
</HTML>
```

运行结果如图 7-2 所示。

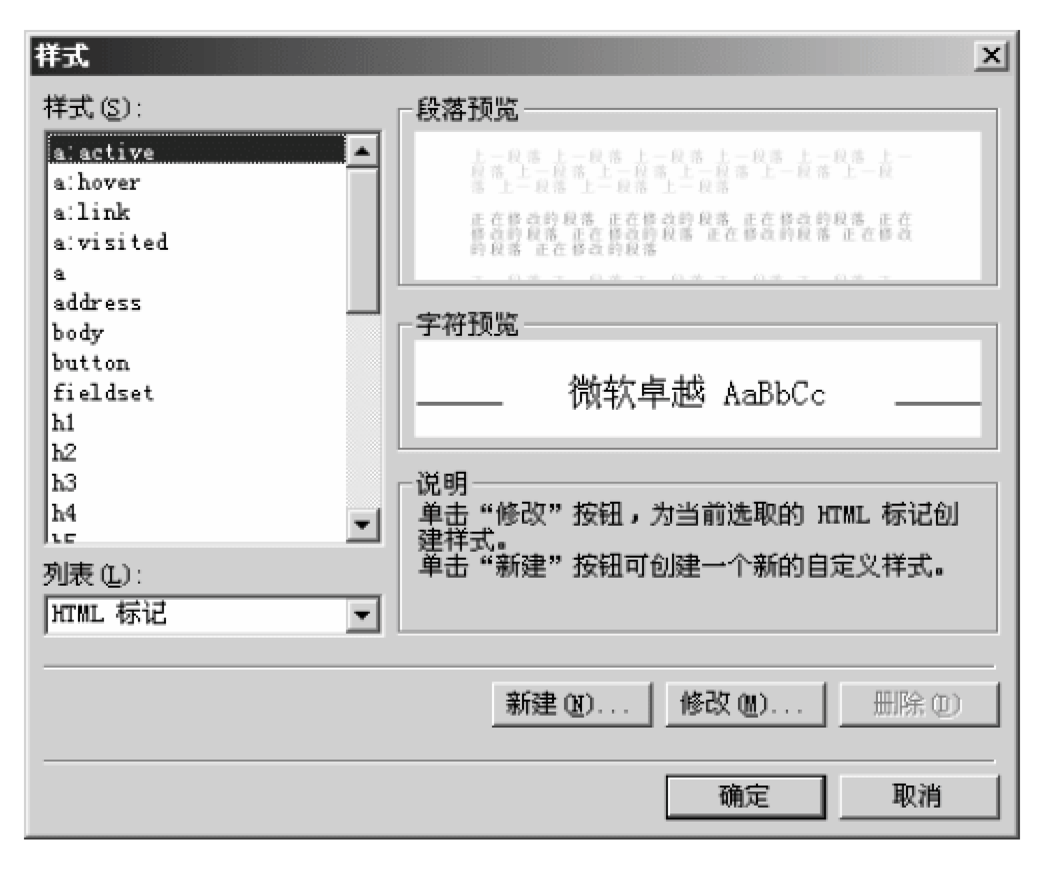





图 7-1　【样式】对话框　　　　图 7-2　内部样式表的使用

【说明】若 FrontPage 中某个标记要应用样式，需要进入代码视图进行文字输入设置。

3．外部样式表

将样式定义保存到一个以.css 为后缀的文件中，然后在每个需要用到这些样式的网页中引用这个 CSS 文件。例如，将以下样式保存到文件 my.css 中：

```
H1.mylayout {border-width: 1; border: solid; text-align: center;color:red}
```

然后在网页的头部使用 link 标记链接到外部样式文件，代码如下：

```
<HTML>
<HEAD>
<link href="my.css" rel="stylesheet" type="text/css">
</HEAD>
<BODY>
<H1 class="mylayout"> 这个标题使用了 Style。</H1>
<H1>这个标题没有使用 Style。</H1>
</BODY>
</HTML>
```

【注意】

（1）CSS 文件中不包括<style>标记，<style>标记是 HTML 标记，而不是 CSS 样式标记。

（2）外部调用 CSS 时要注意文件的链接路径。

在 FrontPage 中创建样式表文件，选择“文件”→“新建”命令，然后选择“其他网页模板”，将得到如图 7-3 所示的对话框，选择“样式表”选项卡。单击“普通样式表”可进入样式表文件的编辑视图。按照前面介绍的内部样式表的创建过程进行样式的设计，建好的样式将存储在样式表文件中。

在 FrontPage 中，若某个网页要引用外部样式，可选择“格式”→“样式表链接”命令，将弹出如图 7-4 所示的对话框。在该对话框中添加选取文件。



图 7-3　外部样式表文件创建模板选择对话框



图 7-4　链接样式表的选取对话框

使用外部样式表的优点如下：

（1）样式代码可以复用。一个外部 CSS 文件可以被很多网页共用。

（2）便于修改。如果要修改样式，只需要修改 CSS 文件，而不需要修改每个网页。

（3）可以提高网页显示的速度。如果样式写在网页中，会降低网页显示的速度；如果网页引用一个 CSS 文件，则该 CSS 文件多半已经在缓存区中（其他网页早已引用过它），网页显示的速度就比较快。

4．样式定义选择的几点注意事项

（1）如果为少量标记添加 CSS，可以采用行内定义的内嵌样式；如果只为一个网页文件使用的样式，则可以采用在文档头定义的内部样式表；如果为众多文件使用的样式，则可以定义外部样式表，通过外部链接使用。

（2）若文档内同时存在多种加入样式的方式，则内嵌样式优先考虑，在<head>标记处定义的内部样式次之，外部样式表优先级别最低。

（3）为了方便理解 CSS 代码，可以写 CSS 代码注释。CSS 代码注释以/*开头，以*/结束。例如：

```
/* 段落样式 */
p{ text-align: center;   /* 居中显示 */
   color: black;
   font-family: arial
}
```

7.3　CSS 属性

CSS 属性可分为字体属性、文本属性、颜色和背景属性、方框属性、分类属性和定位属性等部分。

7.3.1　字体属性

字体属性用于设置字体的名称、大小、显示风格等。

- font-family：定义字体名称，如.s1 {font-family: 宋体}。
- font-size：定义字体大小。有多种单位表示，最常用的就是 pt 和 px(pixel)。
- font-style：定义字体风格。取值有 normal、italic、oblique，其中，normal 是默认值，italic 和 oblique 均为斜体显示。
- font-weight：定义字体浓淡，取值有 normal（正常）和 bold（加粗）。
- font-variant：定义变化字体，取值有 normal 和 small-caps，normal 是默认值，small-caps 表示小的大写字体。
- line-height：定义字体的行高。其值可以按单位值和百分比进行设置。在字体的综合写法中为可选项，用斜杠符与 font-size 分隔。
- font：各种字体属性的一种快捷的综合写法。

各项的排列顺序和格式为[<font- style>||<font-variant>||<font-weight>]?<font-size>[/<line-height>]?<font-family>。

例如，.s1 {font:italic normal bold 11pt arial}表示为样式 s1 定义的字体选用 Arial，风格为斜体，浓淡为加粗，大小为 11pt，font-variant 为 normal。

【例 7-2】　字体属性使用演示

```
------------------------------------------------------------ex7-2.htm------------------------------------------------------------
<HTML>
<HEAD>
<TITLE>字体属性用法</TITLE>
<STYLE type=text/css>
.s1 {font:italic normal bold 16pt arial}
.s2 {font-family:黑体;font-size:24pt;font-weight:bold;color:red}
.s3 {font-family:黑体;font-size:24pt;font-variant:small-caps}
</STYLE>
</HEAD>
<BODY>
<P class=s1>abcd 字体演示 1</P>
<P class=s2>abcd 字体演示 2</P>
<P class=s3>abcd 字体演示 3</P>
</BODY>
</HTML>
```

运行结果如图 7-5 所示。

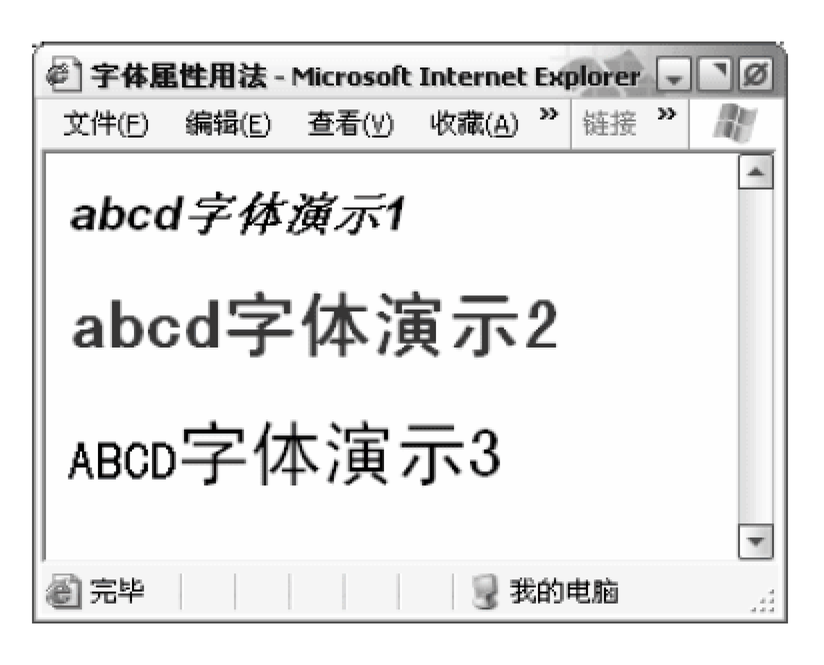


图 7-5 字体属性

【说明】本例在同一段文字上用不同的风格进行显示，读者可对照效果检查具体设置。以下代码通过脚本改变页面上某个层中的字体显示大小。

【例 7-3】 通过脚本代码控制页面元素的字体大小

```
------------------------------------------------------ex7-3.htm------------------------------------------------------
<script language="JavaScript" >
  function setFontSize(f,size) {   //脚本要执行的函数定义
     var obj=document.getElementById(f);
     obj.style.fontSize=size;
  }
</script>
<!--   控制字体大小的超链接   -->
【 <A href=javascript: setFontSize('text','15px')大</A>
<A href=javascript: setFontSize('text','13px')>中</A>
<A href=javascript: setFontSize('text','11px')>小</A>  】
<!--以下为字体大小受控制的页面元素-->
<div id='text'> ……<div>
```

【说明】单击超链接执行脚本函数 setFontSize，改变 id 标识指定元素的字体大小。

7.3.2 文本属性

文本属性包括设置文本的对齐方式、修饰、缩进以及行高和间距等。

- text-align：定义文本对齐方式。取值有 left（居左，默认值）、right（居右）、center（居中）、justify（两端对齐）。
- text-decoration：定义文本修饰。取值有 none（无，默认值）、underline（下划线）、overline（上划线）、line-through（中间划线）。
- text-indent：定义文本缩进。默认值为 normal，其值有按长度和百分比两种设定方法。按长度设置可以用绝对单位（cm、mm、in、pt、pc）或相对单位（em、ex、px），相对单位指相对 normal 情形的增减值；按百分比设置是相当于父对象宽度的百分比，父对象就是外层标记。
- line-height：定义行高。默认值为 normal，其值有按长度和百分比两种设定方法。

- letter-spacing：定义字间距。默认值为 normal，其值有按长度和百分比两种设定方法。
- text-transform：定义文本的变换。取值有 none（无，默认值）、capitalize（首字母大写）、uppercase（字母转大写）和 lowercase（字母转小写）。

【例 7-4】　文本属性样式的使用

```
-----------------------------------------------------ex7-4.htm-----------------------------------------------------
<HTML>
<HEAD><TITLE>文本属性用法</TITLE>
<STYLE type=text/css>
H2.space { LETTER-SPACING: 5pt }                    /*  字符间距  */
P.ind {
     COLOR: #CC6600; TEXT-INDENT: 20pt;
     BACKGROUND-COLOR: #BEFACA;
     font-size:24pt; font-family:华文仿宋; font-weight:bold
}
H3.dec { TEXT-DECORATION: line-through }            /*  删除线  */
P.hei2 { LINE-HEIGHT: 32pt   }                      /*  演示行间距  */
SPAN.super { VERTICAL-ALIGN: super }                /*  上标的使用  */
</STYLE>
</HEAD>
<BODY>
<H2 class=space>本行字符间距是 5pt</H2>
<P class=ind>本段文字起始缩进 20pt</P>
<H3 class=dec>本行文字带有删除线。</H3>
<P class=hei2>本行与下一行间距为 32pt，<BR>本行的 X 和 Y 带有上标：X<SPAN
class=super>3</SPAN>+Y <SPAN class=super>3</SPAN> </P>
</BODY>
</HTML>
```

运行结果如图 7-6 所示。

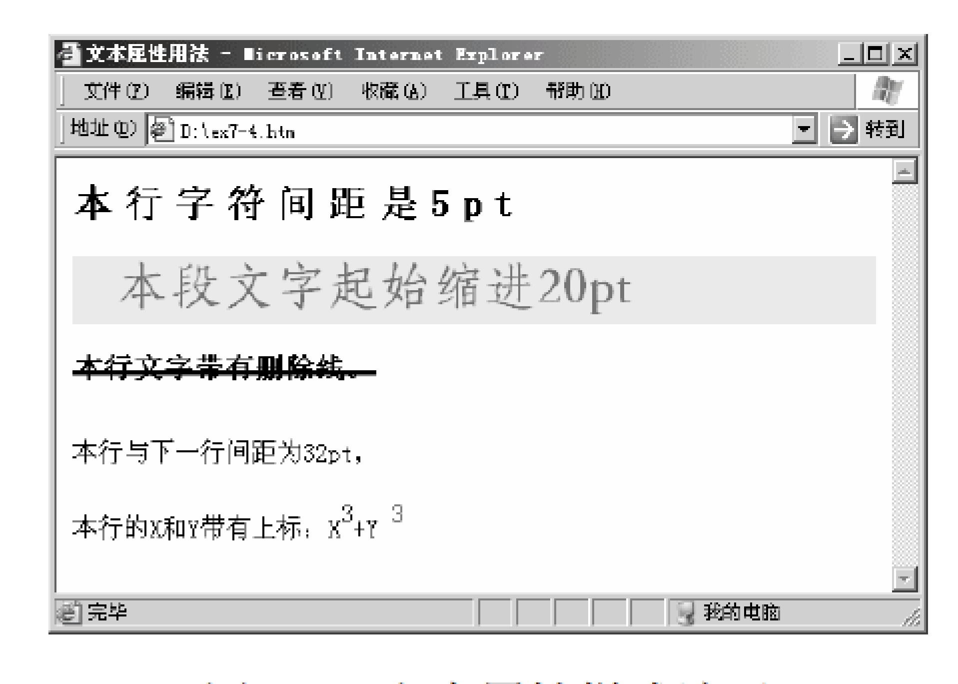


图 7-6　文本属性样式演示

7.3.3　颜色和背景属性

- background-color：定义背景颜色。
- background-image：定义背景图片。
- background-repeat：与 background-image 属性一起使用，设置背景的重复。其中

repeat 为默认值，沿水平和竖直两个方向重复；repeat-x 表示背景图片横向重复；repeat-y 表示背景图片竖向重复；no-repeat 表示背景图片不重复。

- ❑ background-attachment：与 background-image 属性一起使用，决定图片是跟随内容滚动还是固定不动。有两个值，一个是 scroll（滚动），另一个是 fixed（固定）。默认值是 scroll。
- ❑ background-position：与 background-image 属性一起使用，决定背景图片的最初位置。有按百分比定位，也有按偏移像素值定位。默认值为 0%，该值代表左上角为起始位置。

【提示】在 Firefox 和 Opera 中需要把 background-attachment 属性设置为"fixed"，只有这样才能保证该属性正常工作。

以下为背景设置的简单举例：

```
body
{ background-color:#99FF00;background-image:url(background.jpg);background-repeat:no-repeat;
background-position:20px 60px}
```

表示背景图片的初始位置距离网页最左端 20px，距离网页最上端 60px。当背景图片不存在时以背景颜色设置背景。

- ❑ background：背景相关属性的快捷综合写法。例如：

```
body {background:#99FF00 url(background.jpg) no-repeat fixed 40px 100px}
```

以下样式可使页面的背景图案在文字“滚动”时静止不动。

```
<style type="text/css">
BODY { background: purple url(bg.jpg);
        background-repeat:repeat-y;
        background-attachment:fixed
  }
</style>
```

不仅整个网页存在背景，网页的众多元素，如表格、单元格等均可设置背景样式。另外，在后面的样例中还将看到用背景填充层、列表项的情形。

7.3.4 列表属性

列表属性用于设置列表标记（<ol>和<ul>）的显示特性，包括类型、位置、列表图片等。

- ❑ list-style-type：定义列表样式类型。取值有 disc（默认值，黑圆点）、circle（空心圆点）、square（小黑方块）、decimal（数字排序）、lower-roman（小写罗马字排序）、upper-roman（大写罗马字排序）、lower-alpha（小写字母排序）、upper-alpha（大写字母排序）、none（无列表项标记）。
- ❑ list-style-position：定义列表样式位置。取值有 outside（以列表项内容为准对齐）、inside（以列表项标记为准对齐）。

- ❑ list-style-image：定义列表样式图片。例如：

```
ul{list-style-image: url(../images/circle.gif)}
```

- ❑ list-style：为列表样式的快捷综合写法。例如：

```
ul{list-style:circle inside url(images/flow.gif)}
```

如果想无任何风格，则可用 list-style :none。

7.3.5　边框、边距和间隙属性

1．边框属性

边框属性用于设置边框的宽度、风格、颜色。设置方式较多，上下左右 4 个边框既可以统一设定，也可以分开设定。

（1）设置边框宽度的属性

border-top-width、border-bottom-width、border-left-width、border-right-width 属性用于设置元素上、下、左、右边框的宽度，具体取值可通过常量或长度单位决定。常量有 medium（默认值，中等）、thin（细）、thick（粗），长度单位可以用绝对单位（cm、mm、in、pt、pc）或相对单位（em、ex、px）。

（2）设定边框风格

通过 border-style 属性设置，取值有 none（没有边框）、hidden（隐藏）、dotted（点线）、dashed（破折线）、solid（直线）、double（双线）、groove（凹槽）、ridge（突脊）、inset（内陷）、outset（外突）。

（3）设定边框颜色

border-top-color、border-bottom-color、border-left-color、border-right-color 属性用于设定上、下、左、右边框的颜色。

（4）边框属性的快捷综合写法

边框属性还给出了边框设置的快捷综合写法，它包含用 border-width、border-style 和 border-color 分别设置边框的宽度、风格和颜色。

例如，以下定义 img2 的样式设置：

```
.img2 {
    border-style:double;
    border-color:red;
    border-width:3px;
}
```

快捷综合写法可以简写成：

```
.img2{ border：3px red double; }
```

另外，还可以通过 border-top、border-bottom、border-left 和 border-right 分别设置上、下、左、右边框的宽度、样式和颜色。

2．边距属性

边距属性用来设置页面中一个元素所占空间的边缘到相邻元素之间的距离。具体属性有 margin-top、margin-bottom、margin-left、margin-right，它们分别用来设置上、下、左、右边距。例如：

```
.d1{margin-left:1cm}
```

另外，还提供有 Margin 属性为设定边距宽度的快捷综合写法。例如：

```
.d1 {margin:1cm 2cm 3cm 4cm}
```

设置上边距为 1cm，右边距为 2cm，下边距为 3cm，左边距为 4cm。

3．间隙属性

间隙属性（padding）用来设置元素内容到元素边界的距离。具体属性有 padding-top 、margin-bottom、padding-left、padding-right，它们分别用来设置上、下、左、右间隙。属性的取值如表 7-1 所示。

表 7-1　padding 属性的取值情形

取　值	含　义
auto	由浏览器计算间隙
length	规定以具体单位计算的间隙值，如像素、厘米等。默认值为 0px
%	规定基于父元素的宽度的百分比的内边距
inherit	规定间隙从父元素继承

另外，还提供有 padding 属性为设定间隙宽度的快捷综合写法。

以下为用综合写法设置间隙的一些例子，单位均为 px。

- padding:10px 5px 15px 20px;表示上、右、下、左的间隙为 10、5、15、20。
- padding:10px 5px 15px; 表示上、右、下、左的间隙为 10、5、15、5。
- padding:10px 5px; 表示上、下的间隙为 10，左、右的间隙为 5。
- padding:10px; 表示上、下、左、右的间隙均为 10。

CSS 中还有一个重要的概念，即盒子模式，如图 7-7 所示。盒子中由外至里依次是边距（margin）、边框（border）、间隙（padding）、内容部分（有 width 和 height 属性）。

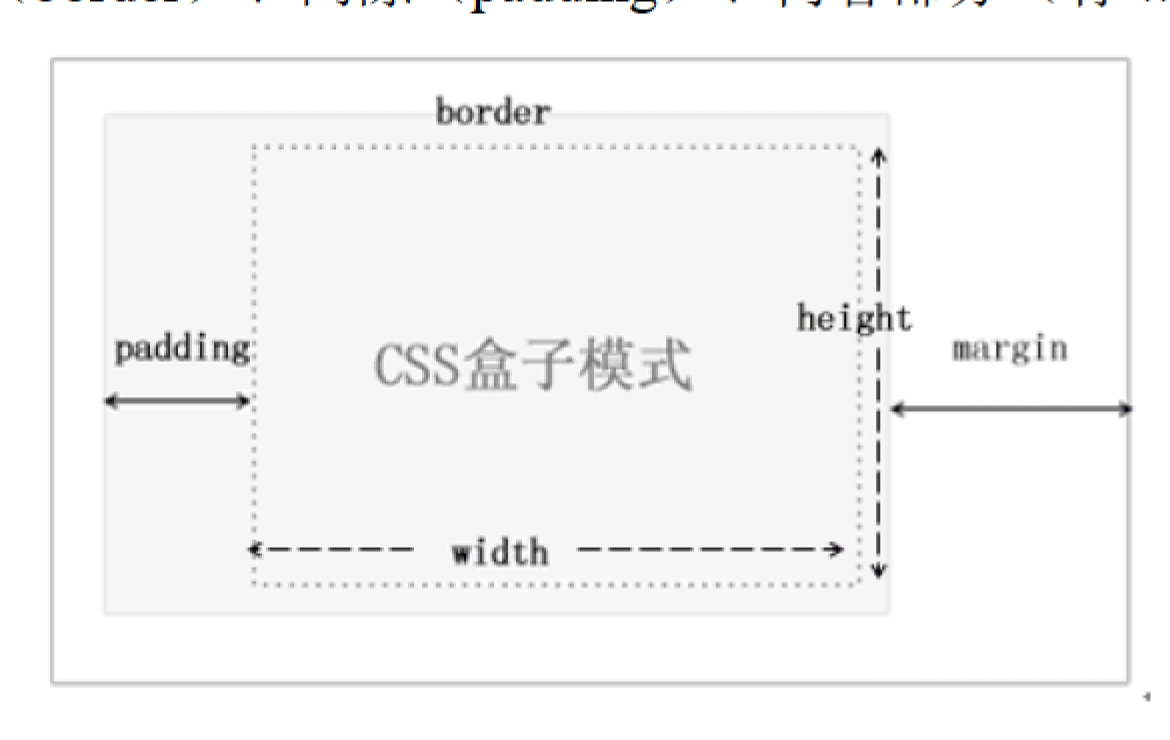


图 7-7　CSS 盒子模式

边距属性（margin）是元素所占空间的边缘到相邻元素之间的距离；边框属性（border）设定一个元素的边线；间隙属性（padding）设置元素内容到元素边框的距离；背景属性指的是内容（虚线框住的部分）和 padding 区域。元素的 width 和 height 属性指的是内容区域的宽和高。

【例 7-5】　一个通用的报错页面设计

```
---------------------------------------------------------error.asp---------------------------------------------------------
<html>
<head>
<style>
body{text-align:center;padding-top:50px;}
#error{ background-image:url(error.png);width:430px;height:240px;
          text-align:left;}
#error p{ font-size:9pt;margin-top:114px;
          color:#333;margin-left:120px;width:265px;line-height:22px}
</style>
</head>
<body>
<div id="error"><p><%=request("mess")%></p></div>
</body>
</html>
```

【说明】仔细体会各元素的样式设置，尤其是盒子模式的应用。该程序可用于网站的统一报错界面。在需要报错时，可重定向到该页面，通过 URL 参数传递报错信息，例如：

```
Response.Redircet("error.asp?mess="&"本课程不允许学生上传文件，请教师联系管理员！")
```

对应产生的访问结果如图 7-8 所示。



图 7-8　网站的统一报错页面

7.3.6　定位与布局属性

1．定位属性

CSS 提供了二维和三维空间定位的属性，即 top、left 和 position。可利用它们定位某个

元素的绝对位置或相对于其他元素的相对位置。

（1）top、left 和 position 属性

- top 属性用于设置元素与窗口上端的距离。
- left 属性用于设置元素与窗口左端的距离。
- position 属性用于设置元素位置的模式，它有以下 3 种取值。
  - absolute：绝对位置，原点在所属块元素的左上角。使用 left、right、top、bottom 等属性相对于其最接近的一个有定位设置的父对象进行绝对定位。如果不存在这样的父对象，则依据 body 对象。
  - relative：相对位置，相对于 HTML 文件中本元素的前一个元素的位置。
  - static：静态位置，按照 HTML 文件中元素的先后顺序显示。

position 属性的默认值为 static。top、left 属性通常与 position 属性配合使用。

（2）三维空间定位

CSS 允许在三维空间中定位元素，与之相关的属性是 z-index，它与 top、left 属性结合使用。z-index 将页面中的元素分成多个“层”，可形成多个层“堆叠”的三维效果。z-index 取值为整数，可为正，也可为负，取值越大表示在堆叠层中处于的位置越高，0 是基准值。

【例 7-6】　三维定位属性用法

```
---------------------------------------------------------ex7-6.htm---------------------------------------------------------
<HTML><HEAD><TITLE>三维定位属性用法</TITLE>
<STYLE type=text/css>
SPAN { FONT-SIZE: 18pt }
SPAN.level2 {
    Z-INDEX: 4;   COLOR: #0000ff;
 font-size:36pt
}
SPAN.level1 {
    Z-INDEX: 2;   COLOR: #A0E1F1;
  font-size:36pt
}
SPAN.level0 {
    COLOR:#A5BDA4; font-size:36pt
}
</STYLE>
</HEAD>
<BODY>
<SPAN class=level2 style="position: absolute; left: 30px; top: 19px">
立体字效果</SPAN>
<SPAN class=level1 style="position: absolute; left: 32px; top: 21px">
立体字效果</SPAN>
<SPAN class=level0 style="position: absolute; left: 34px; top: 23px">
立体字效果</SPAN>
</BODY>
</HTML>
```

运行结果如图 7-9 所示，形成了立体字的效果。



图 7-9　定位属性的使用

2．float（浮动）属性

float 属性用于页面元素的布局设置。取值有 none（默认值，对象不浮动）、left（对象向左浮动）、right（对象向右浮动）。float 属性和 position 属性的区别在于，float 是相对定位的，会随着浏览器的大小和分辨率的变化而改变，而 position 则不会。所以在一般情况下采用 float 进行布局，在局部可能会用到 position 进行定位。

【例 7-7】　利用浮动定位实现菜单栏

文件 1：HTML 文件

```
------------------------------------------------------ex7-7.htm------------------------------------------------------
<html>
<head>
<link rel="stylesheet" type="text/css" href="menustyle.css">
</head>
<body>
……
<div id="menu" >
      <ul>
            <li><a href="like_search.asp">个性化推荐</a></li>
            <li><a href="paper_search.asp">学生文章推荐</a></li>
            <li><a href="#" class=current>教学课件推荐</a></li>
            <li><a href="internet_search.asp">Internet 资源</a></li>
      </ul>
</div>
……
</body>
</HTML>
```

文件 2：CSS 样式文件

```
----------------------------------------------------menustyle.css----------------------------------------------------
#menu   { height: 28; }
#menu   ul{
      margin:0;                                    /*将 menu UL 的边距和间隙均置为 0 */
      padding:0;
      list-style-type:none ;
}
```

```
#menu li{
        display:block;
        float:left;                       /* 向左浮动定位*/
        border-right:1px   solid   #ffffff;
        background-image:url('images/button_bg5.gif');
 }
#menu li a{
      display:block;                      /* 定义为块元素，以便背景能完整覆盖列表项 */
      text-decoration:none;               /* 超链接无下划线 */
      line-height:28px;
      padding:0 28px;                     /* 上下间隙为 0，左右间隙为 28*/
}
#menu li a:hover{
      color:red;
      background-position: 0% 0%;         /* 背景从列表项块的左上方开始*/
      background-image:url('images/button_bg.gif');
}
#menu li a.current{
      background-position: 0% 0%;
      background-image:url('images/button_bg.gif');
}
```

【说明】本例将每个列表项采用向左浮动布局，如图 7-10 中右边的上方一排菜单所示。第一个列表项布署在层 menu 的最左边，第二个列表项顺序在其右边排列，这样依次排列形成美观的选项菜单。

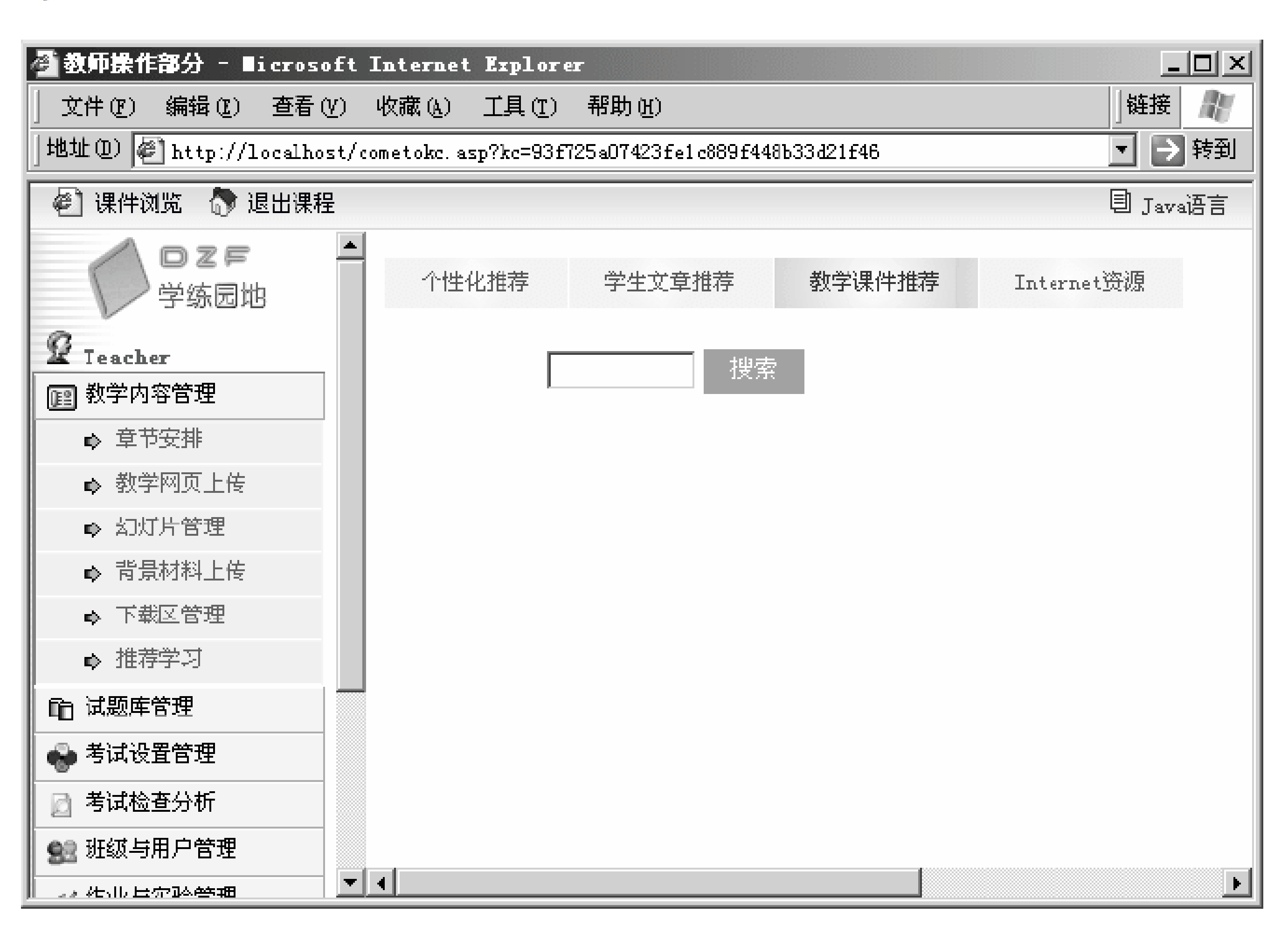


图 7-10　浮动定位的应用（右下框架中菜单项）

3．设置可见性

设置可见性有 display 属性和 visibility 属性。以下为 display 属性的 3 个常见值。

- ❑ block：块对象的默认值。
- ❑ none：隐藏对象。与 visibility 属性的 hidden 值不同，不能为被隐藏的对象保留空间，也就是说页面上其他内容将填充隐藏元素所占的空间。

- inline：内联对象的默认值。

网页的可视文档对象分为块对象（block）和内联对象（inline）。例如，div 是一个块对象，span 是一个内联对象。块对象的特征是从新的一行开始且能包含其他块对象和内联对象。内联对象显示时不会从新行开始，它只能容纳文本或其他内联对象。block 对象的高度、行高以及顶和底边距都可控制；inline 对象的高度、行高以及顶和底边距不可改变。例 7-7 中将列表项（li）定义为 block 对象，这样背景填充时可占满项目空间。

在网页中可通过脚本代码隐藏某个元素。例如：

```
someid.style.display='none'。
```

最后要指出的是，CSS 还包含其他一些属性，读者可通过阅读相关书籍进行了解，限于篇幅，这里不再展开介绍。

本 章 小 结

CSS 样式在网页设计中应用广泛。本章介绍了 CSS 样式的定义与使用的相关知识，包括样式表的种类、样式的定义与引用方式，并结合实例介绍了典型 CSS 属性的应用形式。

习　　题

1．选择题

（1）要将下面代码的超链接文本显示为红色，不能使用的样式表是（　　）。

```
<div><a href="http://www.w3.org/">链接到 W3C</a></div>
```

A．a:link{ color:red }　　B．div a:link{ color:red }
C．div>a:link{ color:red }　　D．div:first-child{ color:red }

（2）使用 CSS 设置格式时，p em{color:blue }表示（　　）。
A．p 元素内的 em 元素为蓝色　　B．p 元素内的元素为蓝色
C．em 元素内的 p 元素为蓝色　　D．em 元素内的元素为蓝色

（3）下列选项中能够实现层的隐藏的是（　　）。
A．display:false　　B．display:hidden
C．display:none　　D．display:" "

（4）下列选项中能够产生带有正方形项目的列表的是（　　）。
A．list-type: square　　B．type: 2
C．type: square　　D．list-style-type: square

（5）下列选项中能够去掉文本超链接的下划线的是（　　）。
A．a {text-decoration:no underline}　　B．a {underline:none}

C．a {decoration:no underline}　　D．a {text-decoration:none}

（6）下列选项中能够设置英文首字母大写的是（　　）。

A．text-transform:uppercase　　B．text-transform:capitalize

C．样式表做不到　　D．text-decoration:none

（7）CSS 中的选择器包括（　　）。

A．超文本标记选择器　　B．类选择器

C．标签选择器　　D．ID 选择器

（8）在 CSS 文本属性中，文本对齐属性的取值有（　　）。

A．auto　　B．justify　　C．center

D．right　　E．left

（9）在 CSS 中，盒模型的属性包括（　　）。

A．font　　B．margin　　C．padding

D．visible　　E．border

（10）下面关于 CSS 的说法正确的是（　　）。

A．CSS 可以控制网页背景图片

B．margin 属性的属性值可以是百分比

C．整个 BODY 可以作为一个 BOX

D．可以使用 word-spacing 属性对中文的字间距进行调整

E．margin 属性不能同时设置 4 个边的边距

（11）在 HTML 文档中，引用外部样式表的正确位置是（　　）。

A．文档的末尾　　B．文档的顶部

C．<body> 部分　　D．<head>部分

（12）下列选项中能够在 CSS 文件中插入注释的是（　　）。

A．// this is a comment　　B．// this is a comment //

C．/* this is a comment */　　D．'this is a comment

（13）在以下选项中，可使所有<p>元素变为粗体的正确语法是（　　）。

A．<p style="font-size:bold">　　B．<p style="text-size:bold">

C．p {font-weight:bold}　　D．p {text-size:bold}

（14）设置边框时，要求上边框 10 像素、下边框 5 像素、左边框 20 像素、右边框 1 像素，下面选项中满足条件的是（　　）。

A．border-width:10px 5px 20px 1px　　B．border-width:10px 20px 5px 1px

C．border-width:5px 20px 10px 1px　　D．border-width:10px 1px 5px 20px

（15）如果将两个层排列在同一行中，下列描述不能实现的是（　　）。

A．直接插入两个 DIV 标记，会自动排在同一行

B．指定 DIV 的 position 属性为 absolute，然后将层位置拖放到同一行中

C．指定 DIV 标记的宽，并且指定其浮动方式，当层宽度之和小于外层元素宽度时，会排在同一行

D．使用一个表格，将两个层分别放入一行中的两个单元格内

（16）下列关于元素在网页中叠放顺序的描述，不正确的是（　　）。

A．使用 CSS 属性 Z-INDEX 来实现叠放顺序

B．属性的取值越大，表示放置的层次越高

C．属性取值可以为负整数

D．可以叠放，但叠放顺序不能指定，将由计算机随机产生

2．问答题

（1）总结 CSS 样式的定义与引用方式，并指出其作用次序。

（2）根据图 7-11 及其代码，回答下列问题：

① “这一行是 h3”的字体大小是多少？

② “main 外的 h2”的字体大小是多少？文字是什么颜色？

③ “这一行是 h1,h1 中的.s2”的字体大小是多少？文字是什么颜色？

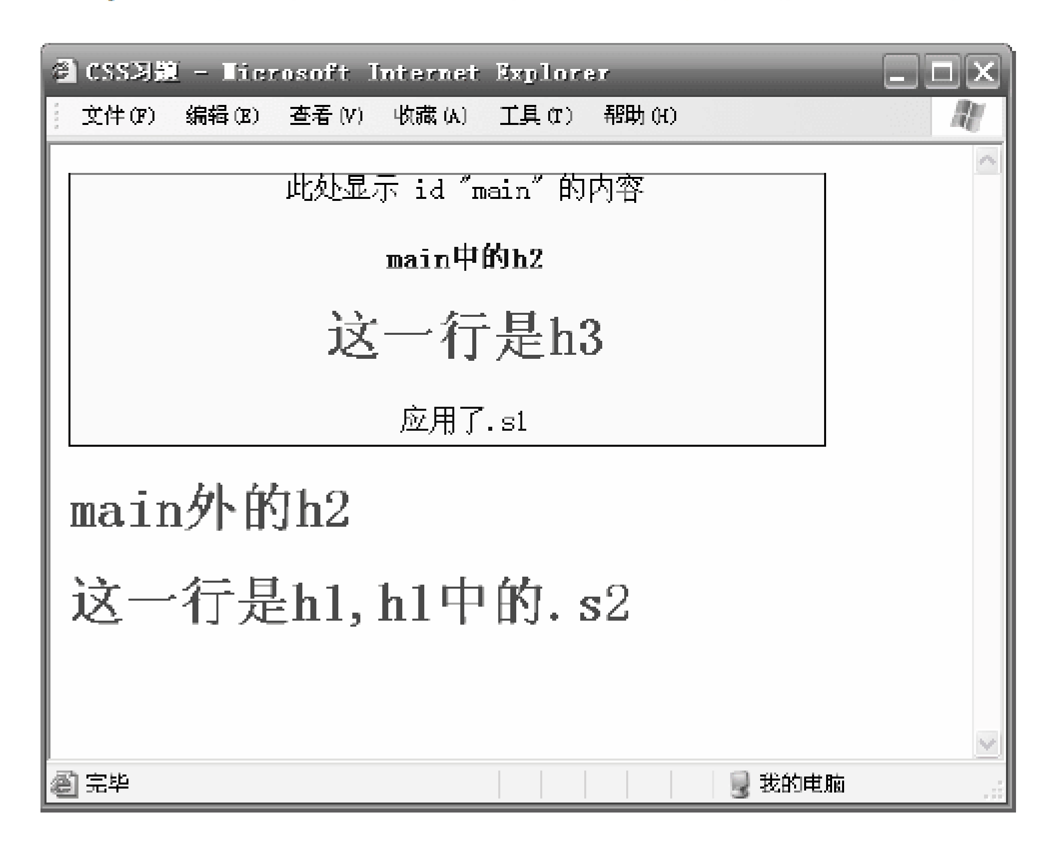


图 7-11　CSS 选择符的应用

图 7-11 对应的代码如下：

```
<html><title>CSS 习题</title>
<style type="text/css">
<!--
#main {
	height: 150px;
	width: 400px;
	border: 1px solid #000066;
	background-color: #ffffcc;
	padding-left: 20px;
	text-align: center;
}
.s1,#main h2 {
	font-size: 16px;
	line-height: 120%;
	color: #0000ff;
}
h3,h2,h1 {
	font-size: 30px;
```

```
        line-height: 110%;
        color: #006600;
}
-->
</style>
</head>
<body>
<div id="main">
   <p>此处显示  id &quot;main&quot; 的内容</p>
   <h2>main 中的 h2</h2>
   <h3>这一行是 h3</h3>
   <p class="s1">应用了.s1</p>
</div>
<h2>main 外的 h2</h2>
<h1>这一行是 h1,h1 中的.s2</h1>
</body>
</html>
```

3．设计题

（1）设计 CSS 样式表，要求网页文字内容显示为 9pt，行距为 150%，颜色为蓝色。表格内文本显示颜色为红色，字体为黑体，大小为 24，行高为 50 像素。超链接样式为：

a:link--黑体 36px，颜色为#333333；

a:hover --黑体 36px，颜色为#FF3300，背景为#CCCCCC；

a:active --黑体 36px，颜色为#999999，背景为#333333；

a:visited--黑体 36px，下划线为“无”，颜色为黑色。

把这个样式表保存为 style1.css，建立两个 HTML 文件（page1.htm 和 page2.htm）应用该样式。网页中包含表格、超链接和文字，能通过超链接从 page1.htm 跳转到 page2.htm。

（2）图 7-12 所示为网络教学中课程的教学幻灯片文件列表页面。利用层的样式定义完成图中的方框界面，标题部分的背景图片由课程教学网站下载，该背景图片的高和宽很小，用背景填充层时只在 x 方向重复。幻灯片的 URL 文件名自己假定。样式包括颜色、文字大小、边框修饰等，超链接要去掉下划线。

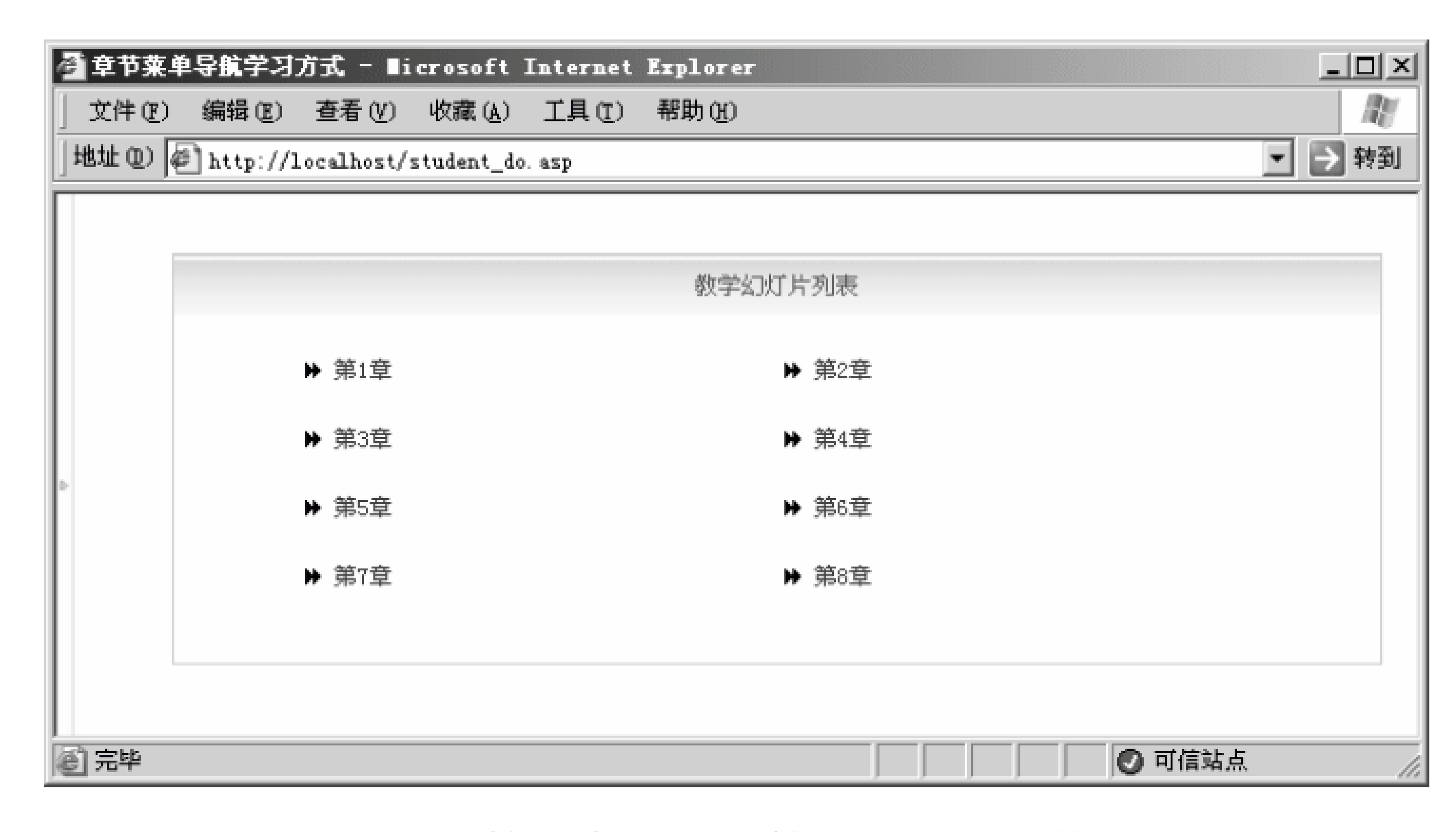


图 7-12　网络教学中课程的教学幻灯片文件列表页面

第 8 章　DHTML 编程

8.1　浏览器对象模型

在 4.0 版以后的 IE 或 Netscape 浏览器都是对象化的，浏览器本身就由许多对象所组成，这些对象有各自的属性、方法和事件。因此，网页设计者可通过脚本程序来控制或调用这些对象。目前，Microsoft 和 Netscape 的浏览器对象模型都是以 W3C 所公布的文档对象模型（Document Object Model，DOM）为基础，再加上扩展对象而成的。

文档对象模型是用于表示 HTML 元素以及 Web 浏览器信息的一个模型，它提供一套表示文档结构的对象以及访问这些对象的方法。DOM 是一种 Web 页面的层次或树型结构表示。基于这个层次结构，还可以创建其他对象。对于每一个页面，浏览器都会自动创建 window、document、location、navigator、history 等对象。

JavaScript 除了可以访问本身内置的各种对象外，还可以访问浏览器提供的对象，从而得到当前网页以及浏览器本身的一些信息，并能完成有关的操作。浏览器常用对象的依托关系如图 8-1 所示。

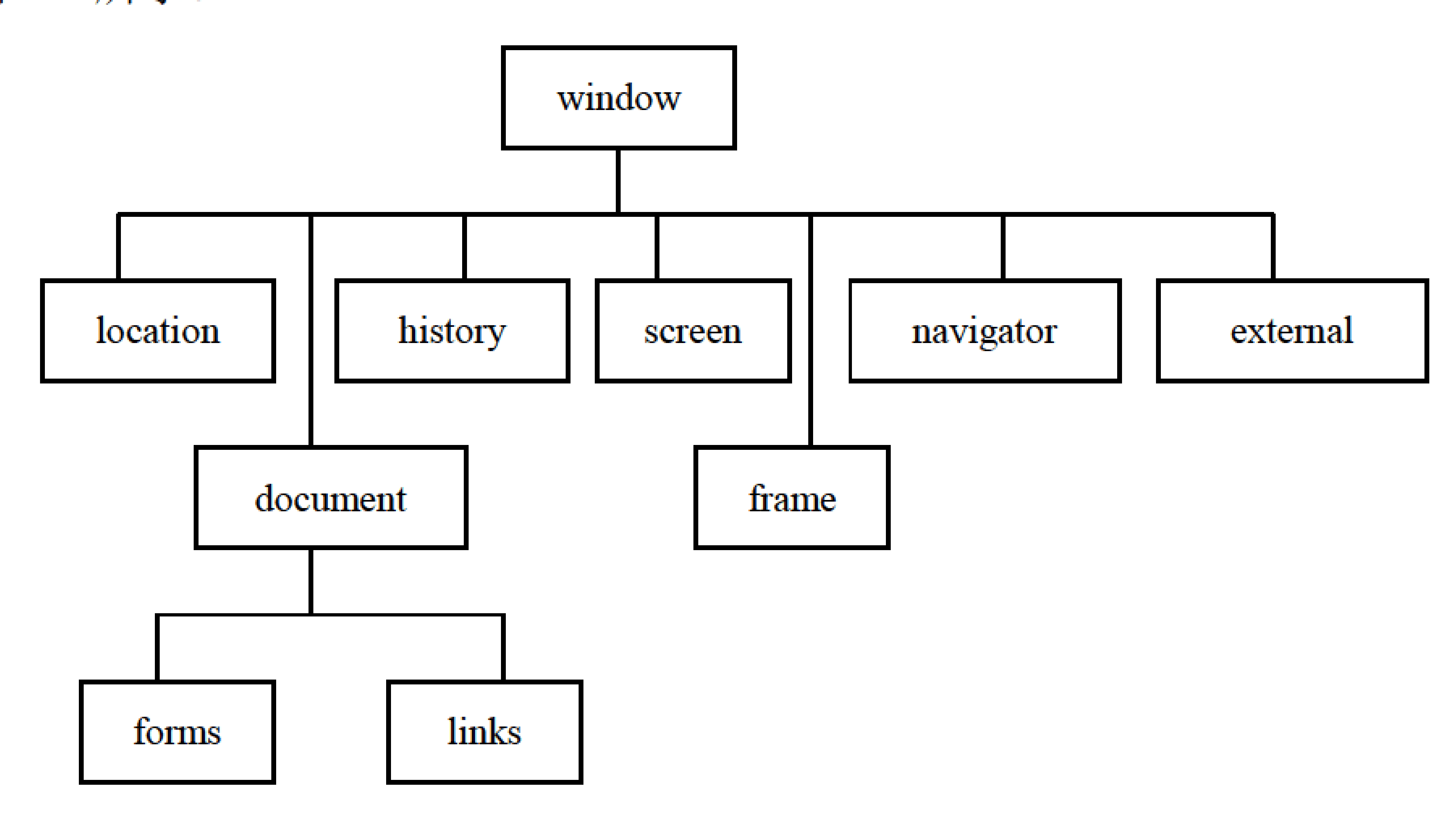


图 8-1　浏览器常用对象的依托关系

8.1.1　window 对象

window 对象描述浏览器窗口特征，它是 document、location、history 等对象的依托对象。

1．window 对象的属性

- frames[]：为子帧数组，每个数组元素代表一个框架，按源文档中定义的顺序存放。

frames.length 为子帧个数。

- self：代表当前窗口。
- parent：代表父窗口（当前窗口是其中一个子窗口）。
- top：代表顶层窗口（是所有可见窗口的父窗口）。
- status：为浏览器状态栏上的消息。
- defaultStatus：当 status 无效时，出现在浏览器状态窗口上的默认消息。
- name：窗体的内部名称，用 window.open()方法打开的窗口定义的名字。
- closed：可以判断一个窗口是否已经被关闭。
- opener：通过该属性可以从一个窗口操纵它的父窗口。

当从一个窗口打开另外一个窗口后，子窗口通过 opener 属性访问父窗口，而通过 open()方法的返回值，父窗口可以和子窗口发生联系，这样两个相关的窗口之间就可以实现互操作。

例如，以下例题先建立一个弹出窗口，通过 opener 属性可以获取父窗口的属性信息。

【例 8-1】　弹出窗口的使用举例

文件 1（x.htm）

```
<script language="javascript">
  popup = window.open("tan.htm","popupnav",width=225,height=235,
  resizable=1,scrollbars="auto");
  window.name ="测试 opener 属性";
 </script>
<BODY>父窗体页面</BODY>
```

文件 2（tan.htm）

```
<HTML><BODY>子窗体页面<p>
<a href="javascript:alert(window.opener.name)">看父窗口的名称</a>
</BODY></HTML>
```

2．window 对象的方法

（1）与对话框有关的方法

- alert("message")：功能是弹出一个警告框，在警告框内显示字符串文本。
- confirm("message")：显示含有给定消息的确认对话框（有一个 OK 按钮和一个 Cancel 按钮），如果用户单击 OK 按钮，返回 True，否则返回 False。
- prompt("message")：显示一个提示输入对话框，要求用户根据显示消息给予相应的输入。

【例 8-2】　3 种对话框的使用样例

```
------------------------------------------------------ex8-2.htm------------------------------------------------------
<script language="javascript">
name="";
name=window.prompt("请输入你的姓名：",name);
window.alert(name+"你好!下面要开始考试了!");
if (window.confirm("你确实准备好了吗?")){
```

```
        window.location.href="exam.asp";
};
</script>
```

【例 8-3】　用户登录检查程序的设计

文件 1：登录页面

```
------------------------------------------------------------mylogin.htm------------------------------------------------------------
<CENTER>
<form action="usercheck.asp"   method="post">
<DIV    style="FILTER:  dropshadow(color=#888888,offx=10,offy=10,positive=1);  WIDTH:215px;
HEIGHT: 102px">
<TABLE height="96%" cellPadding=3 width="97%"
        bgcolor="#F1FAFE" border=0 style="border-collapse: collapse">
<TR>
<TD align=center width="100%" colspan="2" bgcolor="#DDF0FF" >
    <p align="center"><font color="#0000FF">用户登录</font></TD>
</TR> <TR>
<TD align=right width="40%" ><big>登录名</big></TD>
<TD align=center width="60%"> <INPUT type="text" size="12" name="username">
</TD>
</TR> <TR>
<TD align=right width="40%" ><big>口  令</big></TD>
<TD align=center width="60%"> <INPUT type="password" size="12" name="password"> </TD> </TR>
</TABLE></DIV>
<p><input border=0   name=enter src="images/btn_login.gif" type=image>
</form>
</CENTER>
```

【说明】该页面设计时将表格（TABLE）嵌套在一个层（DIV）中，通过层的样式设定形成阴影效果。运行结果如图 8-2 所示。该页面提供了一个登录表单，其中包括 username 和 password 两个文本输入框。



图 8-2　登录页面

文件 2：登录验证程序

```
------------------------------------------------------------usercheck.asp------------------------------------------------------------
<%
user=request.form("username")
pass = request.form("password")
```

```
if   instr(user,"'")>0 or instr(pass,"'")>0 then
      response.write " <script>alert('用户名或密码非法 ！');
      window.location.href='mylogin.htm';</script>"
      response.end
end if
file="data.mdb"
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open connstr                     '连接数据库
sql="select * from userTable where userid='"&user&"' and password='"&pass&"'"
'检查用户名和密码是否正确
set rs=Conn.execute(sql)
if rs.EOF   then                      '记录集是否为空
      response.write "<script>alert('用户名或密码错误！'); "
      response.write "history.back();</script>"
else
   conn.close
   response.cookies("username")=user
   response.redirect("mainpage.asp")
end if %>
```

【说明】

① 本例结合了服务器端和客户端两方面的脚本编程技术。在 Response 对象输出的响应信息中含有在客户端执行的 JavaScript 脚本代码。先执行服务器端脚本代码，输出的 HTTP 响应消息中含有的客户端脚本代码将在客户端解释执行。

② 为了防止 SQL 注入攻击，在用户名或密码输入中不允许出现撇号。因此，程序中安排了一条 if 语句进行检查，如果含有撇号，则在响应页面中通过执行 JavaScript 脚本程序弹出对话框指示错误性质，并利用 window.location.href 的赋值重定向到登录页面。这里需注意 response.end 的作用。

③ 正常登录将检查数据库表格中是否有用户名和密码匹配的用户，如果存在，则将用户身份记录在 Cookie 变量中并转向正常页面；否则，将通过执行 JavaScript 脚本程序弹出对话框指示错误，并用 history 对象的 back()方法让用户重新回到登录界面。

【思考】本例在服务器端检查用户名和密码是否合法是否可以在客户方进行？

（2）与窗口打开和关闭有关的方法

❑　open()方法：打开一个窗口，并指定窗口的风格。语法格式如下：

open("URL","WindowName"[,窗口风格]);

其中，URL 指定要打开的文件页面地址，WindowName 指定新窗体的命名，“窗口风格”为可选参数，它可以指定浏览器是否具有 toolbar（工具栏）、location（地址栏）、directories（目录按钮）、status（状态栏）、menubar（菜单条）、scrollbars（滚动条）等，它们可以设置为 yes 或 no，同时窗口风格也可以指定浏览器窗口的 width（宽）和 height（高）。

open 方法可返回一个打开窗口的引用值，以后可用该引用值去代表窗体对象。例如：

```
nw = open("URL","WindowName");
```

- close()方法：关闭窗口，如 nw.close();。

（3）与超时有关的方法

- setTimeout 方法：创建定时器。语法格式如下：

setTimeout("expression", time)

其中，参数 expression 是一个表达式，通常为函数调用；time 是一个整数，单位为毫秒。该函数的作用是经过指定的毫秒后，自动执行 expression 的内容。

- clearTimeout 方法：清除指定的定时器。语法格式如下：

clearTimeout(timeID)

其中，timeID 为 setTimeout 方法的返回标志。

另一组用来进行定时设置与取消的函数是 setInterval 和 clearInterval，它们的用法和参数形态与 setTimeout 和 clearTimeout 完全一致。

【例 8-4】　简单时钟显示控制

```
------------------------------------------------------------ex8-4.htm------------------------------------------------------------
<html>
<head>
<script language="JavaScript">
var timer;
function clock()  {
      var timestr="";
      var now = new Date();
      var hours = now.getHours();
      var minutes = now.getMinutes();
      var seconds = now.getSeconds();
      timestr += hours;
      timestr += ((minutes<10)? ":0" : ":")+minutes;
      timestr += ((seconds<10)? ":0" : ":")+seconds;
      window.document.frmclock.txttime.value=timestr;
      timer = setTimeout('clock()',1000);    //设置定时器
}
function stopit() { clearTimeout(timer); }
</script>
</head>
<body>
<form action="" method="post" name="frmclock" id="frmclock">
<p>当前时间：<input name="txttime" type="text" id="txttime"></p>
<p><input type="button" name="Submit" value="启动时钟" onclick="clock()">
<input type="button" name="Submit2" value="停止时钟" onclick="stopit()"></p>
</form>
</body>
</html>
```

【说明】这里对 clock 函数的调用是一种特殊的循环往复，也就是通过定时器反复调用自己。本例通过按钮的 onclick 事件触发执行脚本函数，注意对文档中显示时间的输入框的访问，可以通过对象的层次路径访问（window.document.frmclock.txttime），也可以直接

通过 id 名访问，因为它在页面上是唯一的，所以可以省略前面的路径信息。

（4）与窗口焦点有关的方法

在多窗口的操作系统中，任何时候只有一个窗口处于“激活”状态，它可以获取用户的输入，也就是获得了焦点，获得了焦点的窗口称为当前窗口。通过如下方法可改变当前窗口。

- focus()：让窗口获得焦点，如 nw.focus();。
- blur()：让窗口失去焦点，如 nw.blur();。

8.1.2　document 对象

document 对象代表当前页面文档，因此，它包含着当前页面的一些信息，包括页面的前景色和背景色以及页面中的表单、锚标、图像等对象。运用 document 对象可以动态访问和操纵页面内容。

页面中的所有对象都与顶级对象 document 有关，整个文档表现为由 HTML 标记的嵌套层次构成的节点树。从节点树的角度来看，文档是节点的集合，节点是文档树的分支和叶子。

1．document 对象的常用属性

（1）bgColor：页面的背景色。

（2）fgColor：页面的前景色，即文本的颜色。

（3）linkColor：页面的超文本链接的颜色。

（4）lastModified：页面文件的最后修改时间。

（5）title：页面的标题。

（6）URL：页面的 URL 地址。

（7）forms[]：表单（form）对象组成的数组，数组中的每一个元素对应于网页中的每一个<FORM>标记，数组元素的顺序对应在 HTML 文件代码中标记出现的先后顺序。

（8）links[]：超文本链接对象组成的数组，数组中的每一个元素对应于网页中的每一个<A>标记，数组元素的顺序对应在 HTML 文件代码中标记出现的先后顺序。

（9）images[]：文档中所有图像对象组成的数组，数组中的每一个元素对应于网页中的每一个<img>标记，数组元素的顺序对应在 HTML 文件代码中标记出现的先后顺序。

（10）anchors[]：文档中所有锚对象组成的数组。

【例 8-5】　通过 document 对象设置和访问页面的属性

```
-------------------------------------------------------ex8-5.htm-------------------------------------------------------
<html>
<body>
<script>
document.bgColor="fefefe";
document.fgColor="green";
document.linkColor="#0088FF";
```

```
document.alinkColor="#0088FF";
document.vlinkColor="#0088FF";
document.write("<h2>通过 document 对象设置页面属性示例</h2>");
document.write("<a href='http://www.ecjtu.jx.cn'>访问华东交通大学</a>");
document.write("<br><br>本页面的 URL 是："+document.URL);
document.write("<br><br>文件最后被修改时间是："+document.lastModified);
</script>
</body>
</html>
```

运行结果如图 8-3 所示。

2．document 对象的方法

- write 方法：用于在文档中写内容。语法格式如下：

write(String1,String2,…);

该方法的参数可以是一个以上的任意个字符串，如果参数不是字符串，则将自动转化为字符串。常用一个参数的形式，将输出内容用+号拼接在一起。例如：

```
document.write("<br><br>你是第"+i+ "个访问者");
```

- writeln 方法：功能同 write 方法，只是在输出内容后加一个换行符。
- open 方法：打开一个已存在的文档或创建一个新文档来写入内容，允许的文件类型包括 text/html、text/plain、image/gif、image/jpeg、x-word/plug-in 等。
- close 方法：用于关闭文档，以后用 document 写入的内容将成为一个新文档。
- getElementById(String)：返回文档中具有特定 ID 标识的对象。
- getElementsByName(String)：返回文档中具有某名称的所有 HTML 元素对象的集合。
- getElementsByTagName(String)：返回文档中符合指定标记名称的所有 HTML 元素对象的集合。

【例 8-6】　用 open 方法打开新窗口并在新窗体页面填写内容

```
----------------------------------------------------------x.htm----------------------------------------------------------
<html>
<head>
<script language="JavaScript">
function createWin(){
      NewWin=window.open("","","width=200,height=200");
      NewWin.document.open("text/html");
      NewWin.document.write("这是新创建的窗口!");
      NewWin.document.close();
}
</script>
</head>
<body>按下按钮可弹出一个窗口<br>
<form><input type="button" value="弹出新窗口" onClick="createWin()"></form>
</body>
</html>
```

运行结果如图 8-4 所示。



图 8-3　用 document 对象访问和设置页面属性

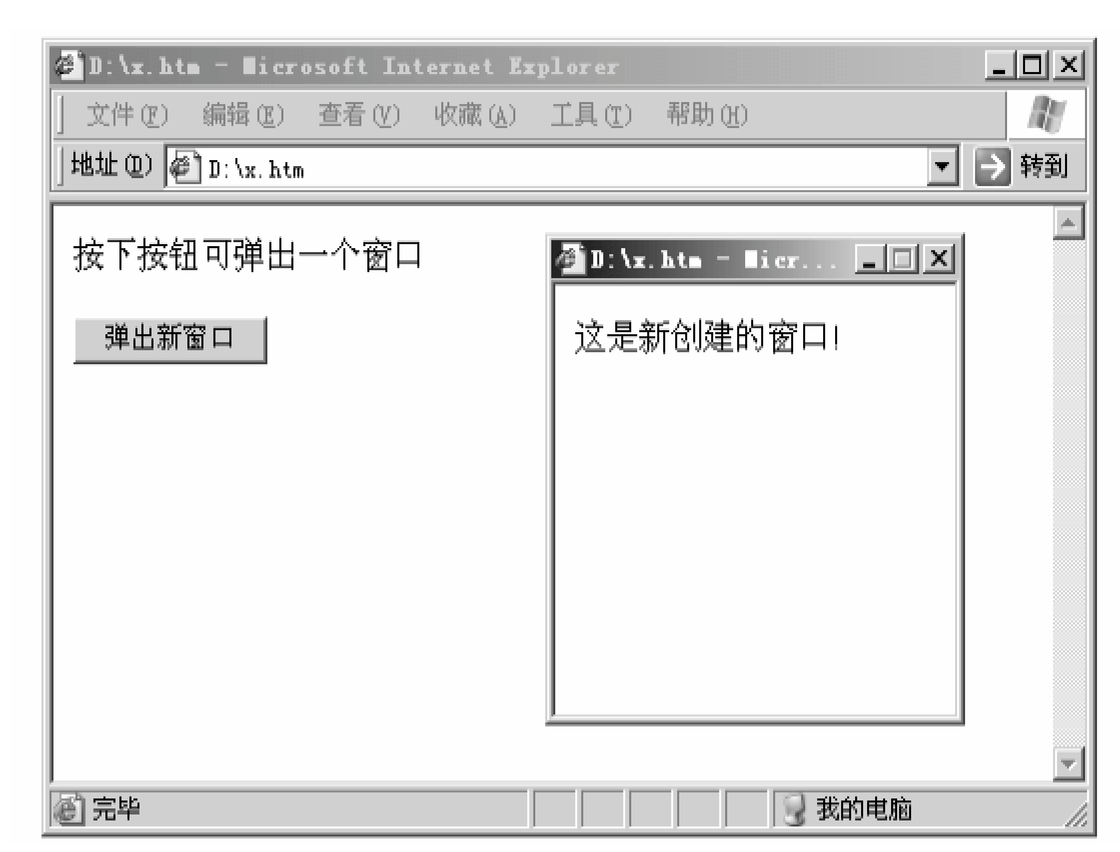
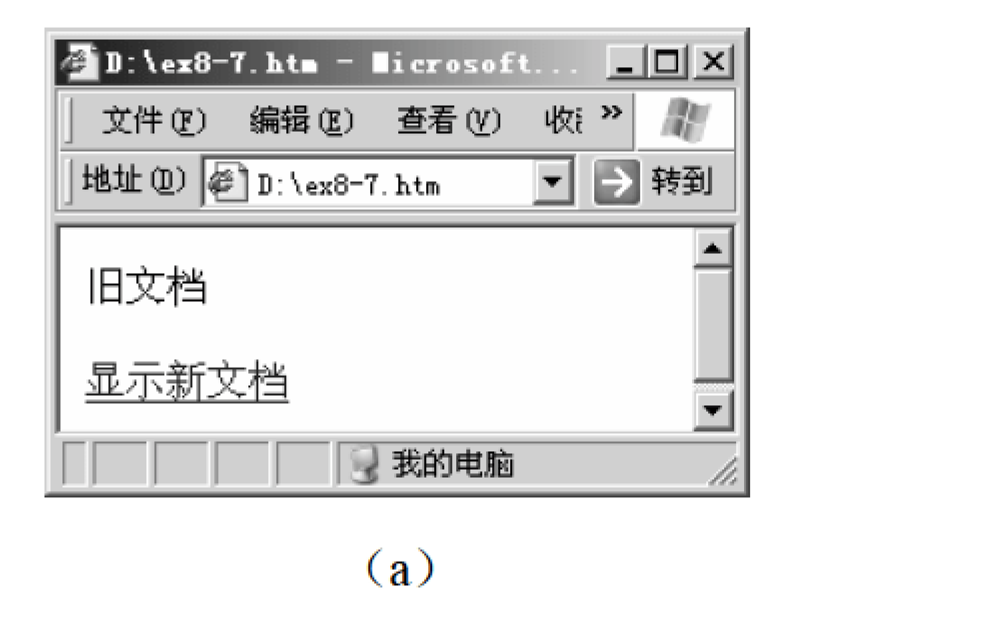


图 8-4　在用 open 方法打开的新窗口中动态书写内容

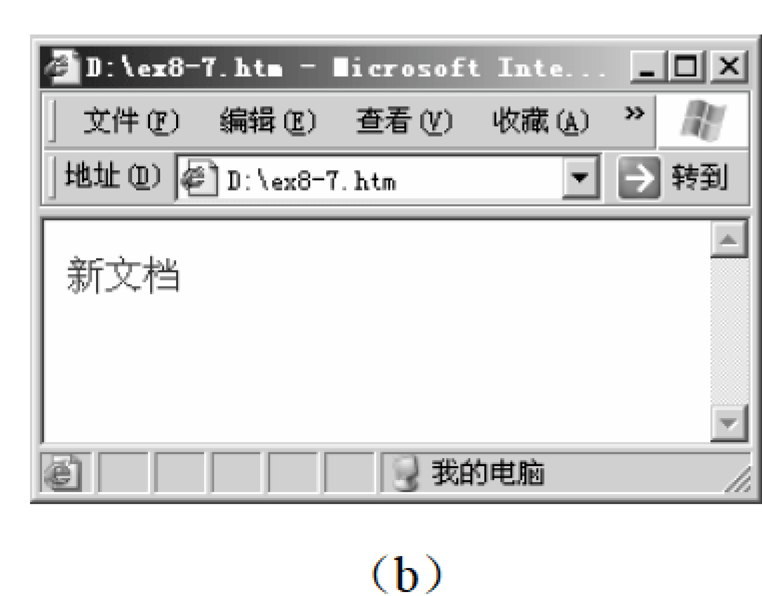
【例 8-7】　改变同一页面文档的内容

```
------------------------------------------------------------ex8-7.htm------------------------------------------------------------
<html>
<head>
<script language="JavaScript">
function change() {
      document.open();   //创建新文档
      document.write("<p><font color=red>新文档</font></p>");
      document.close();
}
</script>
</head>
<body>
<p>旧文档</p><p><a href="#" onclick="change()">显示新文档</a></p>
</body>
</html>
```

运行结果如图 8-5 所示。图 8-5（a）所示为旧文档，图 8-5（b）所示为更改后的页面。



（a）



（b）

图 8-5　动态改变自己页面的内容

3．访问页面文档的元素

文档对象模型将网页中的元素都看作为对象，整个页面体现为由对象元素组成的树型

结构，为了便于访问，可以为页面中需要特别访问的元素规定标识。通常用 id 属性或 name 属性为元素添加标识，在一个网页中每个 id 值应是独一无二的。在脚本中可以通过元素 ID 直接引用元素，也可以通过一个层次路径访问元素，层次路径的一个典型表示为 DOCUMENT.ALL.元素 ID.属性。如果元素中有一个 id 名，则用 getElementById(id 名)是最简单的方法。

【例 8-8】　计算表达式的值

```
--------------------------------------------------------ex8-8.htm--------------------------------------------------------
<html>
<head>
<script language="javascript">
function calculate() {
      frmcalc.result.value=eval(frmcalc.expression.value);
}
</script>
</head>
<body>
<form name="frmcalc" method="post" action="">
```

请输入要计算的表达式：

```
<input name="expression" id="expression" type="text" />
<br>
```

表达式的值为：

```
<input name="result" id="result" type="text" />
<br>
<br>
<input type="button" value=" 计 算 "  onclick="calculate( )"/>
</form>
</body>
</html>
```

【运行示例】用浏览器打开 html 文件，结果如图 8-6 所示。在文本框 expression 中输入任何表达式，单击“计算”按钮，在文本框 result 中将可以看到计算结果。



图 8-6　计算表达式值

【思考】直接通过 ID 访问对象，程序应如何修改？用 getElementById 方法又应如何写？

下面介绍 DHTML 页面设计中常用的两个页面标记。

（1）<span>标记

<span>标记是随着 CSS 的产生而引入的标记，增加该标记的目的是给出样式而不必将样式附加到原有 HTML 标记上，另外，也可以通过更改标记的 style 属性改变样式。以下代码定义了一个 id 为 a2 的 span 标记。

```
<span id="a2"></span>
```

用如下语句可向该标记中写入内容。

```
a2.innerText = XXX
```

或者

```
a2.innerHTML = XXX
```

所有 HTML 元素都有 innerHTML 属性和 innerText 属性。innerHTML 与 innerText 之间的区别在于，innerHTML 是指元素标签间的所有内容（含 HTML 代码），而 innerText 是指元素标签间的文本内容。

（2）<div>标记

<div>标记用来表示层，是一个非常灵活的标记。<div>是块级元素，它将其包含的内容形成一个独立的段落，并且可以嵌套。通过改变层的 style 属性可以让页面内容发生各种效果的动态变化；通过改变层的 innerHTML 属性可改变层的显示内容。

8.1.3　location 对象

该对象包含有当前网页的 URL（统一资源定位器，即网址），其属性 href 用于指定导航到的网页。例如：

```
<body>
<a href=#  onClick="Javascript:window.location.href='nextpage.asp'">按此处到下一个页面</a>
</body>
```

8.1.4　history 对象

该对象包含有最近一段时间访问过的网页的 URL 地址列表。

1. history 对象的属性

- current：历史中当前访问项的 URL。
- next：下一个历史项的 URL。
- previous：上一个历史项的 URL。
- length：历史表中的项数。

2. history 对象的方法

- ❑ back()：装载历史表中的前一个 URL。
- ❑ forward()：装载历史表中的后一个 URL。
- ❑ go(i)：当 i 为数值时，表示装载历史表中与当前项相距 i 的 URL；当 i 为字符串时，表示装载历史表中含字符串的 URL。

【实用举例】在文章阅读页面设计中，首先是在一个页面中显示文章列表，单击某文章标题则可以通过超链接显示文章的具体内容。要从显示文章内容页面返回到前一个页面，可通过设定如下标记实现，这是利用浏览器的回退功能来实现快速返回到先前的页面。

```
<span onclick="history.back( );">返回</span>
```

为了美化界面设计，文字“返回”的位置也可以用图片代替。

【例 8-9】　在网上讨论区更改用户昵称

```
---------------------------------------------------ex8-9.asp---------------------------------------------------
<%
file = "data.mdb"
strpath="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
set conn = server.createobject("adodb.connection")
conn.open strpath                              '连接数据库
user= request.cookies("username")              '取得用户身份
sql="select usercall from user_call where userid='"&user&"'"
'查询用户昵称表
set rs= conn.execute(sql)
if request.form("nicheng")<>"" then            '判断是否属于表单提交处理
  if rs.eof then                               '检查用户是否有昵称
    sql="insert into user_call values('" &user&"','"&request("nicheng")&"')"      '添加用户昵称
  else
    sql="update user_call set usercall='"&request.form("nicheng")&"' where userid='" &user&"'"
'更改用户昵称
  end if
  conn.execute(sql)
  response.write "<script>alert(' 昵称更改成功! ');history.go(-2);</script>"      '回退到讨论页面
  response.end
else                                           '非表单提交，读取用户原有昵称以便进行显示
  if not rs.eof then
     nicheng=rs("usercall")                    '读取原来的用户昵称
  else
     nicheng=user                              '以前无昵称，则默认昵称为用户标识名
  end if
%>
<HTML>
<head>
<STYLE type=text/css>
body{
     COLOR:blue; FONT-FAMILY: "宋体"; FONT-SIZE: 12pt
```

```
}
</STYLE>
</head>
<BODY ><CENTER><br><br>
<form  action="change_call.asp" method="post" >
<fieldset style="width: 80%; height: 204; border: 1px solid #9FB7F4">
<legend>更改用户昵称</legend><br><br>
<TABLE  width="80%" border="0" ><tr>
<TD  align=right  width="53%"  ><font  color="#006666">新昵称(</font><font  color="#FF0000">
*</font><font color="#006666">)</font> </TD>
<TD align=left width="50%">
<INPUT type="text" size="12" name="nicheng" value='<%=nicheng%>'>
</TD> </TR>
</TABLE>
<p align="center">
<INPUT type="submit" value=" 更   改 " style="width: 100; height: 30; color: #008000; background-
image: url('images/button_bg5.gif')" ></p>  </fieldset>
</form>
</BODY>
</HTML>
<%end if%>
```

【使用示例】网络教学系统讨论区用户昵称更改界面如图 8-7 所示。



图 8-7　网络教学系统讨论区用户昵称更改界面

【说明】本例是网络教学系统中更改用户昵称页面的完整程序。在讨论区页面提供了一个超链接进入此程序页面，这是一个自提交的 ASP 程序。更改昵称写入数据库后通过如下输出在页面上显示一个对话框，用户单击对话框后，将执行后面的 history.go(-2)方法回退到讨论页面。

```
response.write "<script>alert(' 昵称更改成功! ');history.go(-2);</script>"
```

【注意】本例中“表单控件分组”和“更改”按钮应用了内联样式。

8.1.5　external 对象

该对象有一个常用的 addFavorite 方法，利用该方法可实现将指定的网页添加到收藏夹中。例如，在页面中设置以下超链接，单击超链接可将网页收藏。

```
<a href=#  onClick="Javascript:
window.external.addFavorite('http://cai.ecjtu.jx.cn/','华东交通大学网络教学平台')">点击珍藏</a>
```

8.1.6　navigator 对象

该对象用于确定用户访问时使用的 navigator 版本，主要有以下属性。

（1）appCodeName：返回用户浏览器的代码名。

（2）appName：返回用户浏览器的实际名字。

（3）appVersion：返回用户浏览器的版本号。

（4）userAgent：该属性反映用户浏览器的全部信息。

8.1.7　screen 对象

screen 对象包含客户端显示屏的有关信息，主要有以下属性。

（1）availHeight：返回不含任务栏的显示屏高度。

（2）availWidth：返回不含任务栏的显示屏宽度。

（3）height：返回显示屏高度。

（4）width：返回显示屏宽度。

（5）pixelDepth：返回显示屏的彩色分辨率（每像素的位数）。该属性 IE 不支持。

例如，某个应用中拟设置两种显示风格，一种是低分辨率（显示宽度小于 1024）的，另一种是高分辨率的。可用如下代码进行检测，并将结果记录到表单的一个隐含域中，以便在后续 ASP 页面中能根据表单提交的值选择页面显示风格。

```
<SCRIPT LANGUAGE="JavaScript">
   if (screen.width <1024) {
        mymode.value="2"            //mymode 为隐含域，1 为高分辨率，2 为低分辨率
      }
</script>
```

8.2　JavaScript 的事件处理

JavaScript 采用了事件驱动的响应机制，用户在网页中的交互操作会触发相应的事件，当事件发生时，系统将自动执行对应的事件处理函数。

8.2.1　JavaScript 事件处理方法

在 JavaScript 中使用事件的第一步是注册事件处理程序，以便浏览器在触发事件时能够运行相应的脚本程序。具体注册方法有多种，其中，内联事件是最常用的方法，直接事件注册也用得比较多，本书的程序实例中主要使用这两种方法，它们可适用于各类浏览器。

在页面上进行某个操作可能导致多个事件发生，处理事件流的技术主要有两种模型：一种是事件冒泡技术，事件从内层元素向外层元素依次触发；另一种是事件捕获技术，事件从外层元素向内层元素依次触发。

在以下 HTML 代码中，p 元素嵌套在 div 元素中。

<div> <p> Event order </p> </div>

假设这两个元素都有 onmouseover 的事件处理程序，如果用户鼠标指针划过该元素，则在事件捕获技术中先触发外层元素 div 的 mouseover；而在事件冒泡技术中，先触发内层元素 p 的 mouseover。IE 支持事件冒泡技术；W3C DOM 既支持事件捕获技术，又支持事件冒泡技术。

1．内联事件注册

通过 HTML 标记的相应属性进行内联事件注册。例如，以下代码通过按钮的 onclick 属性注册单击按钮时执行 startNow()函数。

```
<input id="myid"   type="button" onclick="startNow()">
```

2．直接事件注册

在脚本代码中，使用 HTML 元素的事件属性的定义注册要执行的函数。例如：

```
var myElement = document.getElementById("myid");
myElement.onclick = startNow;                    //注意，这里的函数 startNow 不能加括号
```

在脚本代码中，事件注销用如下方法：

```
myElement.onclick = null;
```

3．为同一事件注册多个处理程序

如果要为元素的一个事件注册多个处理程序，则 IE 和 W3C 的方法各不相同。

（1）IE 事件注册模型的方法

❑　注册事件处理程序

myElement.attachEvent("onclick",fun1);

myElement.attachEvent("onclick",fun2);

其中，fun1、fun2 均为事件处理要调用的函数。

❑　注销事件处理程序

myElement.detachEvent("onclick",fun1);

myElement.detachEvent("onclick",fun2);

（2）W3C DOM 事件注册模型的方法

用 addEventListener 方法进行事件注册，该方法使用 3 个参数，即事件名称、函数名称和一个布尔值。布尔值的设定取决于浏览器的事件处理机制。

❑ 注册事件处理程序

myElement.addEventListener("click",fun1,false)

myElement.addEventListener("click",fun2,false);

其中，如果 addEventListener 的第 3 个参数置为 True，则表示使用事件捕获；如果置为 False，则表示使用事件冒泡。

❑ 注销事件程序

myElement.removeEventListener("click", fun1,false);

若考虑两者的兼容使用，则事件注册的处理方法可进行如下编写：

```
var addH = document.getElementById("myid");
if (addH.addEventListener) {
    addH.addEventListener("click", fun1, false);
} //W3C DOM 注册事件处理程序
else if (addH.attachEvent) {
    addH.attachEvent("onclick", fun1);
} //IE 注册事件处理程序
```

8.2.2　常见事件一览

网页中各类对象的常用事件如表 8-1 所示。

表 8-1　网页中各类对象的常用事件

对 象 类 别	事件处理函数定义位置	事 件 名 称	事件触发条件
网页对象 document	<body>或 document.body	Load Unload ContextMenu Keydown Keypress keyup mousedown …	载入当前网页时触发 试图载入一个新的网页时触发 单击鼠标右键，弹出快捷菜单时触发 按下键盘上的某个键时触发 当用户按下字面按键时触发 当用户释放键盘按键时触发 用任何鼠标键单击对象时触发 …
表单	<form>	Submit Reset	用户提交表单时触发 用户复位表单时触发
文本框 口令输入框 多行文本域 下拉列表框	<input type="text"> <input type="password"> <textarea> <select>	Focus Change Select blur	文本框得到焦点时触发 文本框、下拉列表框内容发生变化时触发 选定文本框中的文本时触发 当控件对象失去焦点时触发

续表

对 象 类 别	事件处理函数定义位置	事 件 名 称	事件触发条件
普通按钮 提交按钮 复位按钮 单选按钮 复选框	<input type="button"> <input type="submit"> <input type="reset"> <input type="radio"> <input type="checkbox">	click	单击按钮时触发
超链接	<a>	click	单击超链接时触发

不同类型的浏览器在事件处理上存在差异，IE 浏览器中以 window.event 对象描述事件。该对象中含有一系列属性，通过访问这些属性，可得到事件相关信息。表 8-2 列出了 window.event 对象的常用属性。

表 8-2　window.event 对象的常用属性

属　　性	含　　义
keyCode	此属性存储击键的代码
x	此属性存储事件相关的 x 坐标
y	此属性存储事件相关的 y 坐标
altKey	此属性当 Alt 键被按下时为真
ctrlKey	此属性当 Ctrl 键被按下时为真
shiftKey	此属性当 Shift 键被按下时为真
button	此属性获取鼠标按键值，左键为 1，右键为 2 或 3，中间键为 4
srcElement	此属性存储产生事件的元素
screenX	此属性存储相对于屏幕实际尺寸的 x 坐标
screenY	此属性存储相对于屏幕实际尺寸的 y 坐标
type	此属性为事件类型的字符串表示

8.2.3　document 的常用事件

1．Load、Unload 事件

Load 事件在网页加载时发生，而 Unload 事件在离开网页时发生。例如，<body　OnUnLoad="alert('谢谢光临本站')">。

在实际应用中经常使用网页装载时执行某个方法来实现网页加载时页面内容的动态显示处理。

【例 8-10】　按比例显示条形图

```
--------------------------------------------------------ex8-10.htm--------------------------------------------------------
<html>
<head>
<script language="javascript">
function calculate() {
    rate=Math.round(Math.random()*100);    //随机产生第一项的显示比例
```

```
    aimg.height = 3*rate;                    //控制图片高度
    p1.innerHTML=rate+"%";                   //显示百分率
    rate=Math.round(Math.random()*100);      //随机产生第二项的显示比例
    bimg.height = 3*rate;
    p2.innerText=rate+"%";
}
</script>
</head>
<body onload="calculate()">
<table>
<td width=50 valign="bottom">
<span id="p1"></span><br>
<img border="0" id="aimg" src="images/bara.gif" width="20">
</td>
<td width=50 valign="bottom">
<span id="p2"></span><br>
<img border="0" id="bimg" src="images/barb.gif" width="20">
</td>
</table>
</body>
</html>
```

运行结果如图 8-8 所示。

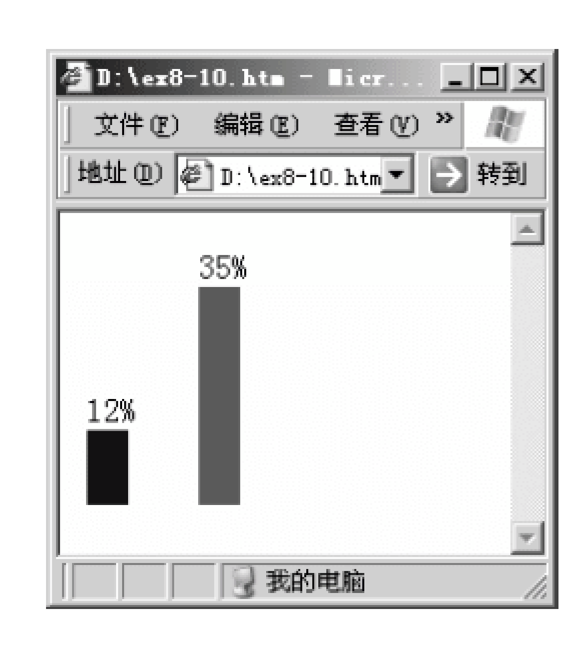


图 8-8　显示条形图

【说明】本例演示了如何利用图片显示高度的调整控制条形图的显示比例，可用于数据分析显示处理。这里利用随机函数产生模拟数据，实际应用中通常需要访问数据库获取分析数据。在程序中直接通过元素标识访问页面对象。

2．ContextMenu 事件

在弹出菜单之前发生该事件，可定义事件过程函数，如果函数返回结果为 True 则弹出，否则不弹出。例如，以下代码直接返回 False，实际上就是禁用了弹出菜单。

```
<body OnContextMenu="return false">
```

3．SelectStart 和 DragStart 事件

通过鼠标拖动来选取网页内容之前发生 SelectStart 事件，可定义事件过程函数，如果函数返回结果为 True 则表示允许选择，否则不能选中。例如，以下是禁止在页面选择内容：

```
<body OnSelectStart="return false">
```

以鼠标拖曳的方式开始选取内容时发生 DragStart 事件。

4．Document 对象的鼠标事件

Document 的鼠标事件有很多，前面两个事件实际也与鼠标有关。以下列出的事件主要是针对鼠标的单击与移动。

（1）Click 事件：单击鼠标时发生。

（2）dbClick 事件：双击鼠标时发生。

（3）MouseDown 事件：按下鼠标左键时发生。

（4）MouseOver 事件：鼠标移到对象上时发生。

（5）MouseOut 事件：鼠标离开对象时发生。

（6）MouseUp 事件：放开鼠标左键时发生。

（7）MouseMove 事件：鼠标移动时发生。

【例 8-11】　借助 mousedown 事件处理禁止用户通过鼠标右键弹出快捷菜单

本例中给出了 document 对象的 mousedown 事件的两种定义办法，处理事件时可以判断按了哪个鼠标键，根据需要进行不同的处理。

```
------------------------------------------------------ex8-11.htm------------------------------------------------------
<script language="javascript">
function nomenu( ){
     if(event.button==2||event.button==3) {
          alert("禁止鼠标右键弹出快捷菜单");
     }
}
document.onmousedown=nomenu;   //定义文档的 mousedown 事件的处理函数
</script>
```

【应用思考】在网页中加入该代码段可禁止用户通过快捷菜单查看源代码，再加上 OnSelectStart 事件可以禁止用户选择文本，在一定程度上实现了网页的防复制。合在一起可写为如下形式：

```
<BODY onSelectStart="return false;" onMouseDown="nomenu( )">
```

5．Document 对象的键盘事件

Document 对象的键盘事件包括 KeyDown 和 KeyPress 事件，当用户按下任意键时首先产生 KeyDown 事件，接着产生 KeyPress 事件，若用户一直按住该键，则会产生连续的 KeyPress 事件。此时若释放按键则产生 Onkeyup 事件。

【例 8-12】　有趣的猜字符练习程序

```
------------------------------------------------------ex8-12.htm------------------------------------------------------
<html>
<head>
<script language="javascript" >
```

```
var pos=0;                                        //当前输入位置
var content="hello…";                             //被猜的内容，实际上可由服务器数据库获取
var s="";                                         //拼接显示结果的字符串
var right=0;
var err=0;
var mylen=content.length;
function display( ){                              //在标识为 x 的层处显示已比对的检查结果
    if (pos < mylen)
      x.innerHTML="<pre>"+s+"<font color=#990099>_</font></pre>";
     else
       alert("得分: "+Math.round(right/(right+err)*100));
}
function mycheck( ){
   if (pos < mylen) {
      c=content.charAt(pos);
      pos++;
      me = String.fromCharCode(event.keyCode);    //获取用户输入的字符
      if (me == c) {                              //将用户输入的字符与答案中的字符进行比较
         s=s+c;
         right++;
      }
      else {
         s=s+"<font color=red>"+c+"</font>";      //输错了，用红色显示
         err++;
      }
    }
    display( );                                   //调显示处理
}
</script>
</head>
<body onKeyPress="mycheck( )" onload="display( )">
<font size=6 color=blue>
<div id=x style="border:3px dotted #008000; padding:10px; width:98%; height:120px">
</div>
</font>
</body>
</html>
```

运行结果如图 8-9 所示。



图 8-9　猜字符练习程序

【说明】该程序通过将用户从键盘输入的字符与标准答案中的字符进行逐个比对，如果输入错误，则用红色显示正确字符。程序中定义了两个函数，即 display()实现内容的显示；mycheck()实现当前位置的字符与键盘输入的比对，并统计对错。

【思考】本例不能处理汉字，因为汉字涉及编码判断。如果练习比对的标准答案为教师提供的模板程序，则此程序改进后可用于编程训练。最好将标准模板程序中的一些内容直接提示给用户，有些让用户填写，此时可通过引入一个标记符来标识模板程序中需要用户填写的部分。读者可思考如何改进本例程序以满足应用要求。

8.2.4　表单处理的常用事件

1．Submit 事件

该事件在表单提交时发生，事件处理函数通常实现表单数据的检验，返回 True 则提交数据，否则禁止提交表单。在以下几种情况下会发生表单提交事件：一是单击了表单的 submit 按钮；二是在表单的文本框中按了回车键；三是单击了表单的图像按钮。

【例 8-13】　表单的自动提交处理

本例可将 ASP 的 Server 对象提供的 htmlEncode 方法转换后的内容进行反转换。如果数据库中的数据在写入时已采用了 Server.htmlEncode 方法进行了变换处理，要恢复成原始的数据可通过该程序。仔细分析该程序，它将转换的数据显示在文本域中再重新提交写入数据库可实现数据恢复。

```
------------------------------------------------------process.asp------------------------------------------------------
<%
x=0+request("n")                    'n 为要处理的记录号，初始请求变量 n 为空，因此 x=0
set conn = server.CreateObject("adodb.connection")
conn.Open "datasource"
strSQL = "select * from mytable"
set rs=server.CreateObject("adodb.recordset")
rs.open strsql,conn,1,2
rs.move x,1                                         '移动到要处理的记录处
if not rs.eof then
   if request("m")<>"" then
      rs("content")=request("m")                    '内容存储在 content 字段中
      rs.update
      conn.close
      response.redirect("process.asp?n="&(x+1))     '转向处理下一条记录
   else %>
      <body onload="javascript:my.submit( );">
      <form name="my" method="post" action="process.asp?n=<%=x%>">
      <textarea name=m cols=80 rows=10 ><%=rs("content")%></textarea>
      </form>
   <% end if
   conn.close
end if
%>
```

【说明】程序借助表单的自动提交实现对所有数据记录的处理。通过 URL 参数传递下一个处理的记录编号，页面装载时通过 onload 事件的代码执行表单的 submit()方法让显示处理表单自动提交。

【应用思考】表单的自动提交处理在网页编程中经常遇到，例如，考试系统中的自动交卷，当考生时间用完时可通过脚本代码让解答表单自动提交。

2．onclick 事件

鼠标单击各类按钮时均发生 onclick 事件，Submit 按钮的 onclick 事件优先于表单的 Submit 事件进行处理。也就是说，如果同时定义了两个事件的处理代码，则先处理 onclick 的事件代码，然后再处理 onSubmit 的事件代码。

【例 8-14】　混合 ASP 和客户端脚本实现网上教学的习题操练

以下程序从数据库读取试题，每屏显示一道试题，利用 ASP 的记录集翻页功能定位当前要显示的试题，此时设定页的大小为 1。通过客户方 JavaScript 脚本判断用户解答的对错，这里编写了一个函数 check(x,m)，其中，参数 x 为一个控件，传递的实际参数是用户选中的解答选项，参数 m 为试题的标准答案。5 个解答选项通过 onclick 属性定义事件发生时执行的 check 函数，本例为 check 函数的参数 x 传递 this 引用，而参数 m 对应的实参从数据库表格代表试题答案的字段 answer 获得。

```
----------------------------------------------------------xztlx.asp----------------------------------------------------------
<SCRIPT LANGUAGE = "JavaScript">
function check(x,m) {                       //此函数检查解答的正确性
  if (x.value==m)  {
     alert(" 对 ! ");
  }
  else  {
     alert(" 错 !");
  }
}
</script>
<%
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open   "connstr"                      //连接数据库
sql="Select * from jvselect "              //选取试题
set rs=Server.CreateObject("ADODB.RecordSet")
rs.open sql,conn,1,1
'以下程序段实现翻动试题
RS.pagesize=1       '每页显示一道试题
ScrollAction = Request("ScrollAction")
PageNo=request("page ")
if ScrollAction <> "" Then
     if ScrollAction = "上一题"     then
          PageNo = PageNo-1
     elseif ScrollAction = "第一题" then
          PageNo = 1
     elseif ScrollAction = "下一题" then
```

```
          PageNo = PageNo+1
     else
          PageNo=rs.pagecount
     end if
     if PageNo < 1 Then
          PageNo = 1
     end if
else
     PageNo = 1
end if
RS.AbsolutePage = PageNo
%>
<center><font color=blue size=3>共有 <font color=red><%=rs.pagecount%></font> 题，此是第
<font color=red><%=PageNo%></font> 题
<!-- 以下显示来自数据库表格的试题内容　　-->
<TABLE>
<TD>
<pre><font size=4><b><%= rs("content")%></b></font></pre>
</TD>
</TABLE>
<!-- 以下表单提供答题界面，当用户选中单选按钮进行答题时将通过 JavaScript 事件执行前面定义
的 JavaScript 函数进行对错检查。-->
<form name="my" method="post">
<table width=60%>
<%  i=65
do while i<=69  %>
<td align=left>
<input type="radio"  name="ans"  value="<%=chr(i)%>"  onclick="check(this,'<%=rs("answer")
%>')"> <%=chr(i)%>
</td>
<% i=i+1
Loop  %>
</table>
</form>
<!--以下表单实现试题翻页处理 -->
<form METHOD="post" ACTION="xztlx.asp" >
<input type="hidden" name="page" value="<%=PageNo%>">
<% if PageNo > 1 Then %>
<input TYPE="SUBMIT"  NAME="ScrollAction" VALUE="第一题">
<input TYPE="SUBMIT"  NAME="ScrollAction" VALUE="上一题">
 <% end if %>
<% if PageNo < rs.pagecount Then %>
<input TYPE="SUBMIT"  NAME="ScrollAction" VALUE="下一题">
<input TYPE="SUBMIT"  NAME="ScrollAction" VALUE="最后一题">
 <% end if %>
<% if rs.pagecount=1 Then %>
<input TYPE="SUBMIT"  NAME="ScrollAction" VALUE="第一题">
 <% end if %>
</form>
```

运行结果如图 8-10 所示。



图 8-10　单选题的交互做题界面

【思考】check 函数的编写也可以考虑改换形式，如将两个参数分别定义为用户解答的值和标准答案，那么函数调用应做相应修改。

由于本例利用翻页显示实现试题的前后浏览，且一屏只显示一道试题，所以也可以考虑直接跳转记录的方式实现试题的翻动。读者可尝试修改程序。

8.2.5　表单中的控件

表单中的各类控件有很多，下面简要介绍几个对象的属性和方法。

1．下拉列表框（select）对象

（1）select 对象的属性

- ❑ length：select 对象中选项的个数。
- ❑ name：select 对象的 name 属性定义的名称。
- ❑ selectedIndex：select 对象中当前被选中的选项的下标。
- ❑ options：select 对象各选项的数组，每个选项有如下属性。
  - ◆ text：对应标记显示的文本串。
  - ◆ value：表示 value 属性的值。
  - ◆ defaultSelected：反映该选项是否为默认选中的布尔值。
  - ◆ selected：反映该选项当前选择状态的布尔值。

（2）select 对象的事件处理器

- ❑ onFocus：定义当对象获得焦点时执行的代码。
- ❑ onBlur：定义当对象失去焦点时执行的代码。
- ❑ onChange：定义当对象的值有所改变时执行的代码。

2．复选框（checkbox）对象

（1）属性

- name：复选框的 name 属性的字符串值。
- value：复选框的 value 属性的字符串值。
- checked：复选框是否选中的布尔值，如果选中，则为 True，否则为 False。
- defaultChecked：反映该选项是否为默认选中的布尔值。

【应用提示】在 JavaScript 脚本代码中，要获取一组复选框的选择值，需要先用 checked 属性检查该复选框是否选中，再通过 value 属性得到相应值。

（2）方法

click()：选中复选框。

（3）事件处理器

onclick：定义用户单击 Checkbox 时执行的事件代码。

3．单选按钮（radio）对象

（1）属性

- name：单选按钮的 name 属性的字符串值。
- length：radio 对象中单选按钮的个数。
- value：选中的单选按钮的 value 属性的字符串值。
- checked：布尔值，按钮被按下时为 True，否则为 False。
- defaultChecked：按钮的 checked 属性的初值，是反映按钮是否为默认选中的布尔值。

（2）方法

click()：选中单选按钮。

（3）事件处理器

onclick：定义当单选按钮被选中时执行的事件代码。

最后要注意的是，页面上的所有元素均可看作为对象，表格中的单元格也可以定义 onclick 属性进行代码编程处理。图片上的鼠标事件在编程处理中应用也很广泛。例如，以下是网络教学系统中功能导航区域的一个图片的鼠标事件处理，在图片的 onmouseover 事件中定义了要执行的 3 个函数。关于具体代码，读者可到该页面查看。

```
<img border="0" src="images/funch6-6.gif"   name=Image3 onmouseover="MM_swapImgRestore();
MM_swapImage('Image3','','images/funch6-m.gif',1);changemenu(4)" >
```

本 章 小 结

本章介绍了 DHTML 技术中的浏览器对象模型以及 JavaScript 的事件处理。这些内容和技术处理是客户端编程的重要组成部分。文档对象模型（DOM）将页面上的所有元素均看作为对象，通过 DOM 结合脚本编程可动态改变页面的内容和显示风格。本章的重点是

掌握 window、document、form 等对象的属性和方法，熟悉典型事件处理。利用本章介绍的内容可以让浏览器页面实现很好的交互性。

习　　题

1．选择题

（1）在 DHTML 中把整个文档的各个元素作为对象处理的技术是（　　）。

A．HTML　　B．CSS　　C．DOM　　D．JavaScript

（2）关于 IE 的 window 对象，下列表述中正确的是（　　）。

A．window.opener 属性本身就是指向 window 对象的

B．window.reload()方法可以用来刷新当前页面

C．window.location="a.html"和 window.location.href="a.html"的作用都是把当前页面替换成 a.html 页面

D．定义了全局变量 g，可以用 window.g 的方式来存取该变量

（3）关于 JavaScript 事件，下列说法正确的是（　　）。

A．事件是用户对浏览器所做的特定的动作（操作），是实现交互操作的一种机制

B．对象发生改变时调用的事件是 onChange

C．当一个表单中的对象被单击时，执行的 JavaScript 事件是 onclick

D．当浏览器加载完成一个窗口或加载完成框架集合中的所有框架时，执行的 JavaScript 事件是 onLoad

（4）JavaScript 的 onSubmit 事件的作用是（　　）。

A．当一个表单中的对象被单击时，执行的是 JavaScript 事件

B．当用户提交一个表单时，需要执行的是 JavaScript 事件

C．当鼠标移出对象时发生的事件

D．对象发生改变时调用的事件

（5）有关 window 对象，下列说法正确的是（　　）。

A．代表一个浏览器的窗口或框架

B．是一个文档、链接或历史对象组的顶层对象

C．窗口对象不能设置状态栏默认信息

D．判断窗口是否关闭，可以使用窗口对象的 closed 属性

（6）有关窗口对象的属性，下列说法正确的是（　　）。

A．设置状态栏的临时信息是用 status 属性

B．查看该窗口最近查阅过的网页用 history 属性

C．对当前窗口进行操作是用 self 属性

D．对最上方的窗口进行操作是用 top 属性

E．要设置浏览器滚动条，可以使用窗口对象的 scrollbars 属性

（7）有关 open 方法的窗口规格参数，下列说法正确的是（　　）。

A．是否显示网址工具栏，用 location

B．是否显示菜单工具栏，用 menubar

C．是否显示滚动条，用 scrollbars

D．是否显示状态栏，用 status

E．是否可以改变窗口的大小，用 resize

（8）关于表单对象的方法，下列说法正确的是（　　）。

A．handleEvent(事件)可以使事件处理程序生效

B．reset()可以重置表单元素

C．submit()可以提交表单

D．表单对象的方法是表单对象为完成需求而调用的方法

E．表单对象的方法是表单对象通过事件而调用的方法

（9）选择对象的属性主要有（　　）。

A．form 表示该对象所在的表单

B．name 表示该对象的 name 属性

C．length 表示该对象的选项的数目

D．options 表示该对象的<option>标记

E．selectedIndex 表示该对象的所选项目的索引值

（10）要获取一个 ID 为 username 的表单元素的值，不正确的代码是（　　）。

A．document.username.value

B．document.all.username.value

C．document.getElementById("username").value

D．如果表单元素外层无表单，则可以直接使用 username.value

（11）如果想控制一个名为 menuBar 的层左移 20 个像素显示，下列能实现的是（　　）。

A．document.menuBar.display += 20

B．document.all.menuBar.pixelLeft += 20

C．document.all.menuBar.left += 20

D．document.all.menuBar.style.pixelLeft += 20

2．设计题

（1）利用脚本代码将网页的状态栏设置为“你的班级+姓名的网页！欢迎你！”。

（2）关闭网页时，为访问者弹出在该网页停留的时间，显示格式为“**分**秒”。注意：不能只显示为几秒，若超过 60 秒则要换成分钟进行显示。

（3）编写个人信息注册表单，利用 JavaScript 代码检查表单输入。若满足如下条件则提交表单，否则不提交。

❑ 用户名不能为空，也不能含有英文单引号字符。

❑ 密码至少要 6 位且两个密码必须相同。

❑ 电子邮件地址中必须包含@符号。

运用以下两种方式编程实现表单提交：

① 利用表单自身的提交功能，在表单提交事件处理代码返回 True 时才正式提交。

② 在按钮单击事件处理代码中调用表单对象的 submit()方法实现表单提交。

（4）参照书中单选题练习的样例实现多选题的交互做题。试题及标准答案存储在数据库表格中。由于每道题的选择可以多个，所以不能依靠选项上发生的事件来判别解答对错，这时可以借助表单提交事件，利用翻动试题按钮触发表单提交。

【提示】利用选项的 checked 属性可检查选项是否选中，然后将各选项的值拼接在一起即为用户解答。

（5）编程实现在网页中轮流显示不同图片的动画效果，安排几个按钮控制该动画的效果，分别为放大、缩小、停止、启动。以下为实现图片更替显示的参考代码：

```
<head>
<Script Language="JavaScript">
var i=1
function image_change( )  {
   if(i==5)  {i=1;}
   my_image.src="Image"+i+".jpg";
   i=i+1;
   setTimeout('image_change( )',1000);
}
</Script>
</head>
<body onload="image_change( )" >
<img src="" name="my_image" width="32" height="32" id="my_image" />
</body>
```

【提示】放大、缩小通过改变图片的高度和宽度实现。

（6）利用 JavaScript 实现一个简单计算器。使用表格安排计算器上的按钮，包括数字按钮、基本算术运算符、等号及清除按钮。

（7）利用 DHTML 技术实现一个三子棋连线的游戏程序。只要横、竖、对角线出现同一方的棋子就算胜。可考虑用表格进行布局，在单元格内单击事件触发下棋。棋子用图片表示，每个单元格要进行标识，已下棋单元格的棋子的颜色用一个数值代表并记录在数组中。每下一步棋进行一次胜负检查。

第 9 章　XML 技术与应用

XML 为扩展标记语言（eXtensible Markup Language），是由 W3C 定义的一种新的 Internet 数据描述和交换标准。XML 在描述数据内容的同时能突出对结构的描述，从而体现出数据之间的关系。在 XML 中用户可以自由地定义标记名以及与标记相关的元素及元素层次。可以说，XML 的出现给数据交换带来了一场革命。本章主要介绍 XML 的典型应用。

9.1　XML 文档格式

XML 文档的基本结构由序言部分和一个根元素组成。序言包括 XML 声明和 DTD（Document Type Define，文档定义类型）或 XML Schema，两者是用来描述 XML 文档结构的，也就是描述元素和属性应遵守的类型约束。

1．元素

元素是 XML 文档内容的基本单元，是由起始标签、元素内容和结束标签组成的。其语法格式如下：

<标签>文本内容</标签>

在 XML 中没有任何保留字，除了必须遵守下列规范外，可以用任何词语来作为元素名称。

（1）名称中可以包含字母、数字及其他字母。

（2）名称不能以数字或“_”（下划线）开头。

（3）名称不能以字母 xml（或 XML 或 Xml ..）开头。

（4）名称中不能包含空格或“:”（冒号）。

无论文本内容有多长或多么复杂，XML 元素中仍可以再嵌套其他元素，这样就可以使相关信息按一定层次结构进行组织。

【例 9-1】　描述职员信息的 XML 文档

```
<employees>
    <employee>
        <name>Lars Peterson</name>
        <salary>25000</salary>
    </employee>
    <employee>
        <name>Charlotte M. Cooper</name>
        <salary>34500</salary>
    </employee>
</employees>
```

【说明】在<employees>的元素中包括了所有职员的信息，每位职员都由<employee>元素来描述，而<employee>元素中又嵌套了<name>和<salary>元素。

如果一个元素从文件头的序言部分之后开始一直到文件尾，包含了文件中所有的数据信息，那么该元素称之为根元素。

从本例中可看出，XML 元素是可以嵌套的，被嵌套在内的元素称为子元素。本例中，<employee>就是<employees>的子元素。

除了元素，XML 文档中能出现的有效对象还包括处理指令、注释和属性。

2．处理指令

处理指令为 XML 解析器提供信息，使其能够正确解释文档内容，它的起始标识是“<?”，结束标识是“?>”。常见的 XML 声明就是一个处理指令：

```
<?xml version="1.0" standalone="yes/no" encoding="UTF-8"?>
```

声明的作用是告诉浏览器或其他处理程序：这个文档是 XML 文档。声明语句中的 version 表示文档遵守的 XML 规范的版本；standalone 表示文档是否附带 DTD 文件，如果有，则参数为 yes；encoding 表示文档所用的语言编码，默认是 UTF-8。为支持中文显示，可在 XML 文档开头加入如下处理指令：

```
<?xml version="1.0" encoding="gb2312"?>
```

处理指令还有其他用途，如定义文档的编码方式是 GB 码还是 Unicode 码，或是把一个样式单文件应用到 XML 文档上用以显示。

3．注释

注释可出现在 XML 元素间的任何位置，但是不可以嵌套，它与 HTML 中的定义形式一样。其语法格式如下：

<!--注释-->

4．属性

属性为元素提供了进一步的说明信息，必须出现在起始标签中。属性以名称/取值对出现，名称与取值之间用等号“=”分隔，并用引号把取值引起来。例如：

```
<salary currency="US$"> 25000 </salary>
```

上例中的属性说明了薪水的货币单位是美元。

一个 XML 文档首先要满足“格式良好”的要求。具体包括以下几点：

（1）根元素唯一。

（2）起始标签和结束标签应当匹配，其中结束标签是必不可少的。空标识是指标识对之间没有内容的标识，其必须关闭，如
、<img>等标识。针对这样的空标识，XML 中的处理方法是在原标识最后加/，如将
写为
。

（3）大小写应一致。XML 对字母的大小写是敏感的，<employee>和<Employee>是完全不同的两个标签，所以结束标签在匹配时一定要注意大小写一致。

（4）元素应正确嵌套。子元素应完全包括在父元素中。<A><B></A></B>就是嵌套错误。

（5）属性值必须包括在引号中。在 HTML 代码中，属性值可以加引号，也可以不加。但是在 XML 中则规定所有属性值必须加引号（可以是单引号，也可以是双引号），否则将被视为错误。

（6）元素中的属性是不允许重复的。

XML 文档的“有效性”是指一个 XML 文档应遵守 DTD 文件或 Schema 的规定，“有效的”XML 文档肯定是“格式良好的”。本章将主要介绍 XML 数据的访问处理，而省略 DTD 和 XML Schema 的介绍，关于 DTD 和 Schema 的介绍，读者可参看相关书籍。

9.2　XML 文档对象模型

文档对象模型（Document Object Model，DOM）是对 Web 文档进行应用开发、编程的应用程序接口（API），是 W3C 公布的一种跨平台的、与语言无关的接口规范。

DOM 采用对象模型和一系列接口来描述 XML 文档的内容和结构，即利用对象把文档模型化。DOM 对结构化的 XML 文档进行解析，文档中的指令、元素、实体、属性等所有个体都可以用对象模型表示，整个文档的逻辑结构类似一棵树，如图 9-1 所示，树的每个结点均代表一个具体对象，每个对象同时包含了方法和属性。

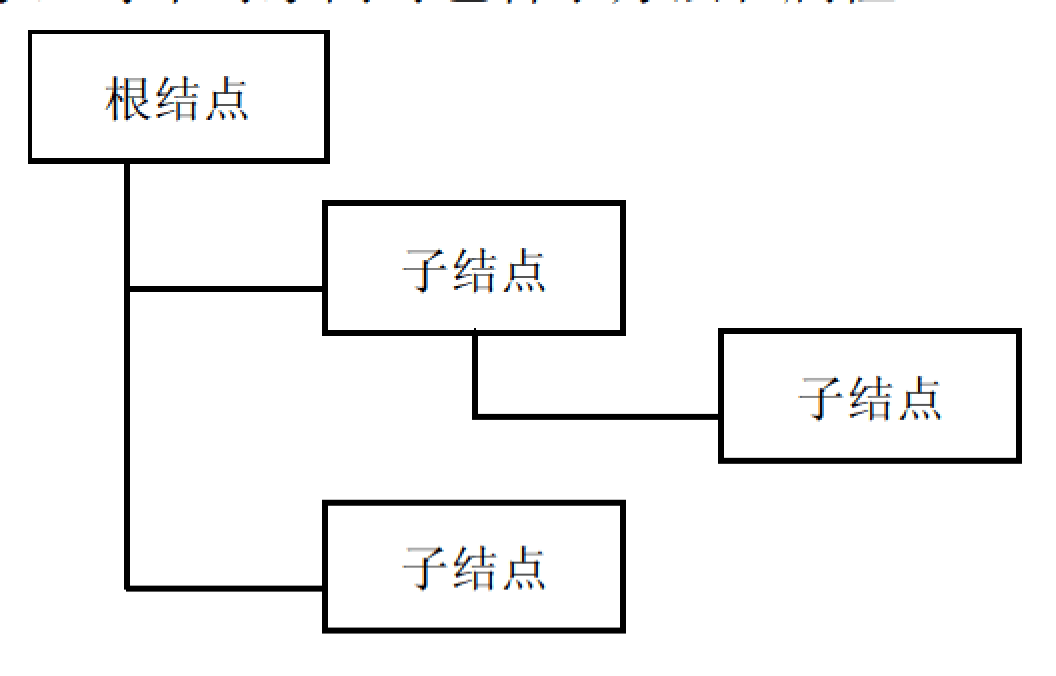


图 9-1　DOM 结点树型层次

利用 DOM，开发人员可以进行动态地创建 XML 文档、结构遍历、添加/修改/删除内容等操作。DOM 具有的面向对象的特性使人们在处理和解析 XML 相关的事务时可节省大量精力，是一种符合代码重用思想的强有力的编程工具。

XML 文档对象模型由以下几种基本的对象组成。

（1）DOMDocument：文档对象，它代表整个文档树的顶级结点，是对整个文档进行操作的入口。

（2）XMLDOMNode：XML DOM 结点树中的一个结点对象。其包括 Element 结点、Attribute 结点、Text 结点等，其中，Text 结点表达了元素或属性的文本内容，它不再包含任何子结点。

（3）XMLDOMNodeList：若干结点组成的集合。

（4）XMLDOMParseError：一个无内容的对象，返回最后一个解析错误的信息。
（5）XMLHTTPRequest：允许和 HTTP 服务器通信。

9.2.1　DOMDocument 对象

DOMDocument 对象是 XML 对象模型中的基础，它所提供的属性和方法可以实现 XML 文件的加载和保存，并可浏览、查询和修改文档树的结构和内容。

1．创建 DOMDocument 对象

在 Web 应用的客户端和服务器端均可通过系统组件创建 DOMDocument 对象。

（1）如果浏览器使用 JavaScript 作为脚本语言，则可用如下代码创建 XML 文档对象：

```
var xmlDoc=new ActiveXObject("Microsoft.XMLDOM")
```

（2）如果浏览器使用 VBScript 作为脚本语言，则可用如下代码创建 XML 文档对象：

```
set xmlDoc=CreateObject("Microsoft.XMLDOM")
```

（3）在 ASP 程序中如果使用 VBScript 脚本语言，则可用如下代码创建 XML 文档对象：

```
set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
```

2．DOMDocument 对象的常用属性

（1）async：表示是否允许异步下载。如果该属性设置为 False，则确保 XML 解析器把 XML 文档完全载入以前不会对 XML 文档进行解析。

（2）attributes：返回目前结点的属性列表。

（3）documentElement：XML 文档的根（Root）结点。

（4）readyState 属性：XML 文件的加载状况。该属性的具体取值说明如表 9-1 所示。

表 9-1　readyState 属性的取值说明

值	意　义	说　明
0	UNINITIALIZED	XML 对象被产生，但没有任何文件被加载
1	LOADING	加载程序进行中，但文件尚未开始解析
2	LOADED	部分文件已经加载且进行解析，但对象模型尚未生效
3	INTERACTIVE	对象模型仅对已加载的部分文件内容有效
4	COMPLETED	文件已完全加载，代表加载成功

（5）onreadystatechange 属性：指定一个事件来处理 onreadystatechange 事件。该事件能辨识 readyState 属性的改变。

3．DOMDocument 对象的方法

（1）加载和存储 XML 文档

Load 和 loadXML 方法可装载解析 XML 内容，并建立 DOM 对象。save 方法可将 DOM

转化为 xml 文本形式存储到文件中。

- load(url)：从指定位置加载 XML 文件。

其中，参数 url 包含要被加载文档的 URL 字符串。若文件加载成功，则返回值为 True；否则返回值为 False。例如：

```
<script type="text/javascript">
var xmlDoc = new ActiveXObject("Microsoft.XMLDOM")
xmlDoc.async="false"
xmlDoc.load("note.xml")   //载入文件名字为 note.xml 的 XML 文档
</script>
```

- loadXML(xmlString)：从一个字符串加载 XML 文本。
- save(filepath)：将 XML 文档以文本形式存储到文件中。

（2）访问 XML 文档树

- getElementsByTagName(tagname)：返回指定名称的元素集合。

其中，参数 tagname 是一个字符串，代表找到的元素卷标名称。使用 tagname"*"传回文件中所有找到的元素。

（3）建立新结点

- createAttribute(name)：建立一个指定名称的属性。
- createElement(tagName)：建立一个指定名称的元素。
- createTextNode(data)：建立一个新的 text 结点，并包含指定的数据。
- createNode(type,tagName,ns)：根据指定的结点类型、名称和名字空间建立新结点。
- setAttribute(name,value)：设置结点的某属性值。

9.2.2　XMLDOMNode 对象

XMLDOMNode 对象表示树中的一个结点，它是 XML 对象模型的最主要对象，元素、属性、注释、处理指令等均能以 XMLDOMNode 来表示。另外，DOMDocument 对象就是继承自 XMLDOMNode 对象，所以这里介绍的方法在 DOMDocument 对象中也能使用。

1．XMLDOMNode 对象的属性

（1）firstChild 属性：结点的第一个子元素。

（2）previousSibling 属性：结点之前的兄弟结点。

（3）nextSibling 属性：结点的下一个兄弟结点。

（4）lastChild 属性：结点的最后子元素。

（5）parentNode 属性：结点的父结点。

（6）nodeName 属性：结点名称。

（7）nodeType 属性：结点类型。例如，1 代表元素（ELEMENT）结点，2 代表属性（ATTRIBUTE）结点等。

（8）nodeValue 属性：用来访问属性和文本结点的值。

（9）xml 属性：结点子树的 XML 描述。

（10）text 属性：结点子树所包含的文字。

（11）childNodes 属性：结点的所有子结点的集合。

【例 9-2】　childNodes 属性的使用

```
--------------------------------------------------------ex9-2.htm--------------------------------------------------------
<script language="javascript">
var xmlobj=new ActiveXObject("Microsoft.XMLDOM");
xmlobj.loadXML("<employees><employee><name>张三</name>
   </employee><employee><name>李四</name></employee></employees>");
var peoples= xmlobj.documentElement.childNodes;
alert(peoples.item(0).text)
alert(peoples.item(1).xml);
</script>
```

运行结果如图 9-2 所示。





图 9-2　输出结点的信息

【说明】通常遍历访问结点树是从 XML 文档的根结点出发，用 XML 文档对象的 documentElement 属性得到根结点，也就是<employees>结点。针对该结点的 childNodes 属性得到的是所有<employee>结点的集合。分别用 text 属性和 xml 属性输出子元素信息，比较两个结果的差异。如果直接用 XML 文档对象（xmlobj）访问 childNodes 属性，则得到的集合中只包括一个根结点元素，即<employees>结点。

2．XMLDOMNode 对象的常用方法

（1）haschildnodes 方法：如果指定的结点有一个或更多子结点，则返回值为 True。

（2）selectNodes(patternString)：所有符合 patternString 样式的结点的集合。如果没有符合样式的结点，则返回空的集合。其中，参数 patternString 为一包含 XSL 样式的字符串。

（3）selectSingleNode(patternString)：返回第一个符合样式的结点。如果没有符合样式的结点，则返回 null。其中，patternString 为一包含 XSL 样式的字符串。

（4）appendChild(newChild)：加上一个结点当作指定结点最后的子结点。

（5）removeChild(oldChild)：将指定的结点从结点清单中移除。其中，oldChild 为一个要被移除的结点对象。

（6）replaceChild(newChild, oldChild)：将原来的子结点以新的子结点取代。

（7）insertBefore(newChild, refChild)：将新结点插入到参考结点的前面。若参考结点省略，则新结点插入到所有子结点的前面。

（8）transformNode(stylesheet)：使用提供的样式表来处理该结点及其子结点。其中，

stylesheet 为一 XML 文件或是片断包含负责结点转换工作的 XSL 元素。此方法会返回一包含转换结果的字符串。

（9）getAttribute(attributeName)：读取结点某属性的值。

以下例子介绍如何将数据库表中的数据保存到 XML 文件中。

【例 9-3】 将数据库中存储的单选题转换为 XML 形式并存储到 XML 文件中

数据库表格（dxt）的字段说明如表 9-2 所示。

表 9-2　数据库表格（dxt）的字段说明

字　段　名	含　　义	对应 XML 标识
point_no	知识点	point
answer	标准答案	answer
content	内容	content

转换后的 XML 文档格式如下：

```
<questions>
  <question>
    <point>1-0-0</point>
    <content>试题内容…</content>
    <answer> A </anwer>
    </question>
  ……
<questions>
```

【说明】根结点为 questions，每道试题为一个 question 结点。此时将每道试题的各字段作为子元素结点，也可以作为属性结点，读者思考如何表示。从这里可看出 XML 标记也可以是中文名称。

以下为用 ASP 编写的转换程序代码，它将从数据库读取试题产生 XML 文件。

```
-----------------------------------------------------ex9-3.asp-----------------------------------------------------------------
<%
set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
xmlDoc.loadxml "<questions></questions>"            '初始文档只有根结点
file="data.mdb"
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Set Conn=Server.CreateObject("ADODB.connection")
Conn.Open connstr                                   '连接数据库
sqlStr = "select * from dxt "
set rs = Conn.Execute(sqlStr)            '从数据库读取试题创建记录集
do while not rs.EOF                      '从记录集读取各个记录的数据添加到 XML 结点树中
  set qNode = xmlDoc. createElement("question")
  xmlDoc.documentElement.appendChild(qNode)    '添加一道试题
  '试题的各个字段作为子结点添加到 question 结点上
  set childNode=xmlDoc. createElement("point")
  childNode.text = rs ("point_no")
  qNode.appendChild(childNode)
```

```
    set childNode=xmlDoc.createElement ("content")
    childNode.text =rs ("content")
    qNode.appendChild(childNode)
    set childNode=xmlDoc.createElement ("answer")
    childNode.text = ""&rs ("answer")
    qNode.appendChild(childNode)
      rs.MoveNext
loop
Conn.close
xmlDoc.save Server.mappath ("exercise.xml")    '保存 XML 数据到文件中
%>
```

【说明】首先利用文档对象的 loadXML 方法并根据 XML 文本生成文档对象的根 questions，然后在此基础上从试题库读取试题产生一个个元素结点并添加到文档中，这里利用 createElement 方法创建元素结点，利用结点的 appendChild 方法将元素结点添加到结点树的对应位置。最后，将生成的 XML 文档保存到文件中。

9.2.3　XMLDOMNodeList 对象

XMLDOMNodeList 是结点集合对象，该对象通常是访问结点的 ChildNodes 属性，或者使用结点的 selectNodes 方法或 getElementsByTagName 方法等返回的结果。

1．XMLDOMNodeList 对象的属性

length 属性是 XMLDOMNodeList 对象的唯一属性，返回集合中项目的个数。

2．XMLDOMNodeList 对象的方法

（1）item(index)：以顺序编号从结点集合中取得某一位置的结点。其中，index 指定子结点的排序位置；0 对应第一个子结点。

（2）nextNode()：存取集合中的下一个结点，若无法取得下一个结点，则返回 null。

9.3　XML 文档的显示处理

XML 文档用来表示数据信息的层次结构，它本身并不包括任何显示信息。如何将 XML 数据在浏览上按特定的要求进行显示有多种方法：最简单的方法是用 CSS 进行显示处理，但因其处理能力有限，所以很少用；最常用的方法是用 XSL 变换进行 XML 文档的显示处理。另外，也常用 XML DOM 访问 XML 文档并结合 DHTML 技术实现显示处理。

9.3.1　利用 CSS 显示

1．示例

【例 9-4】　利用 CSS 显示 XML 文档

（1）XML 文件

```
--------------------------------------------------------student.xml--------------------------------------------------------
<?xml version="1.0" encoding="gb2312" ?>
<?xml-stylesheet type="text/css" href="mystyle.css"?>
<roster>
    学生花名册
  <student>
      <name>李华</name>
      <origin>河北</origin>
      <age>15</age>
      <telephone>62875555</telephone>
  </student>
  <student>
      <name>张三</name>
      <origin>北京</origin>
      <age>14</age>
      <telephone>82873425</telephone>
  </student>
</roster>
```

其中，第二行指定显示样式，如果缺少第二行，则按浏览器默认的风格显示 XML 文件，如图 9-3 所示。

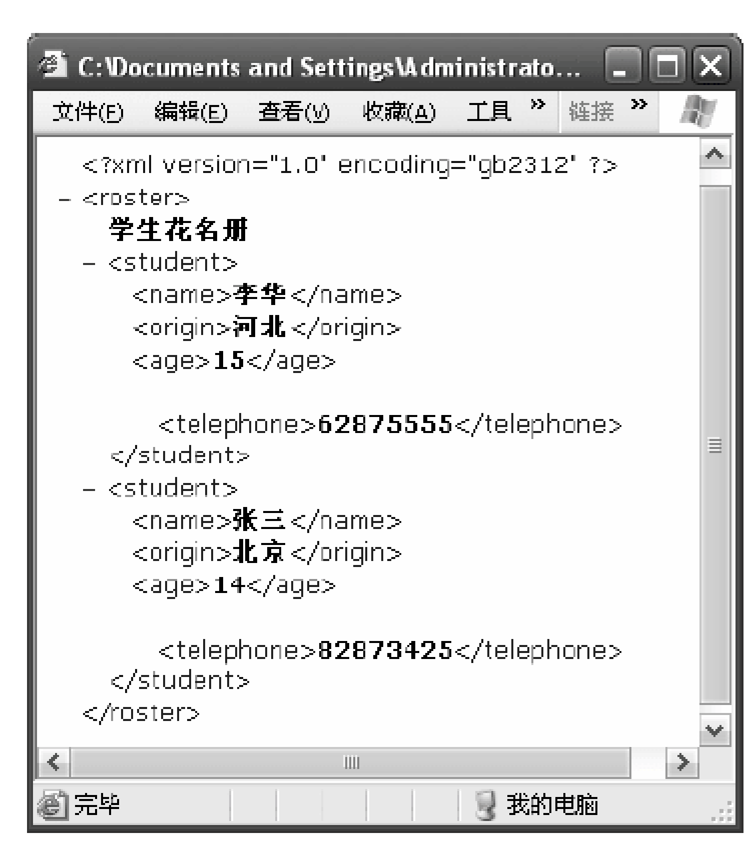


图 9-3　浏览器默认按树型结构显示 XML 文档

（2）CSS 样式文件

```
--------------------------------------------------------mystyle.css--------------------------------------------------------
roster,student
{
   font-size:18pt;
   font-weight:bold;
   color:#008080;              /*  青色  */
   display:block;
   margin-bottom:5pt; font-family:黑体
}
origin,age,telephone
```

```
{
   font-size:12pt;
   display:block;
   margin-left:20pt; text-decoration:underline
}
name
{
   font-size:14pt;
   display:block;
   color:#008000;              /* 绿色 */
   margin-top:5pt;
   margin-left:8pt
}
```

用 360 浏览器浏览 XML 文件，运行结果如图 9-4 所示。



图 9-4　用 CSS 显示 XML 文档

【说明】在 CSS 文档中对 XML 内容中的每个标记定义一个显示样式。在输出结点文本时将自动套用相应的样式。

2．使用 CSS 显示 XML 的缺点

使用 CSS 显示 XML 数据，能力非常有限，主要缺点如下：

（1）XML 的表现依赖浏览器对 CSS 的支持。

（2）只能规定元素而不能规定属性的显示样式。

（3）每个输入元素仅能处理一次。

（4）不能为输出添加元素和其他内容。

（5）不能实现条件和选择处理。

9.3.2　使用 Xpath 查找结点

Xpath 是一个通用的查询表示语法，用于查询和过滤 XML 文件的结点。在 Xpath 语法中可以使用层次结构式的描述方式来找出符合条件的结点。例如，book/author 表示找出所有 book 之下的 author 结点。

所有 Xpath 查询都发生在一个特定的范围内，假设从树的根结点查找所有名为 X 的结点，可能返回一组结果，从树的某一分支查找 X 则可能得到另一组结果。

Xpath 除了在前面 DOM 部分介绍的 selectNodes 方法和 selectSingleNode 方法中使用外，在后面将要介绍的 XSL 样式语言编程中也会用到。

与文件系统的路径标识相似，如果一个查找方式以“/”起始，则表示要以树的根结点及其所包含的结点作为查找范围；如果以“//”起始，则表示查找不限于目前这一层，还可包括其下所有层的结点。

以下结合具体样例介绍 Xpath 的典型使用。设有如下 XML 文档：

```
<Data>
  <item id="1">
    <desc>项目一</desc>
  </item>
  <item id="2">
    <desc>项目二</desc>
    <item id="1">
      <desc>子项目一</desc>
    </item>
  </item>
</Data>
```

此时 item 标记代表一个项目，desc 标记代表项目的描述，因此该 XML 文档的结点树示意图如图 9-5 所示。在该示意图中为了方便描述，不妨为 3 个 item 结点加上 A、B、C 标识。

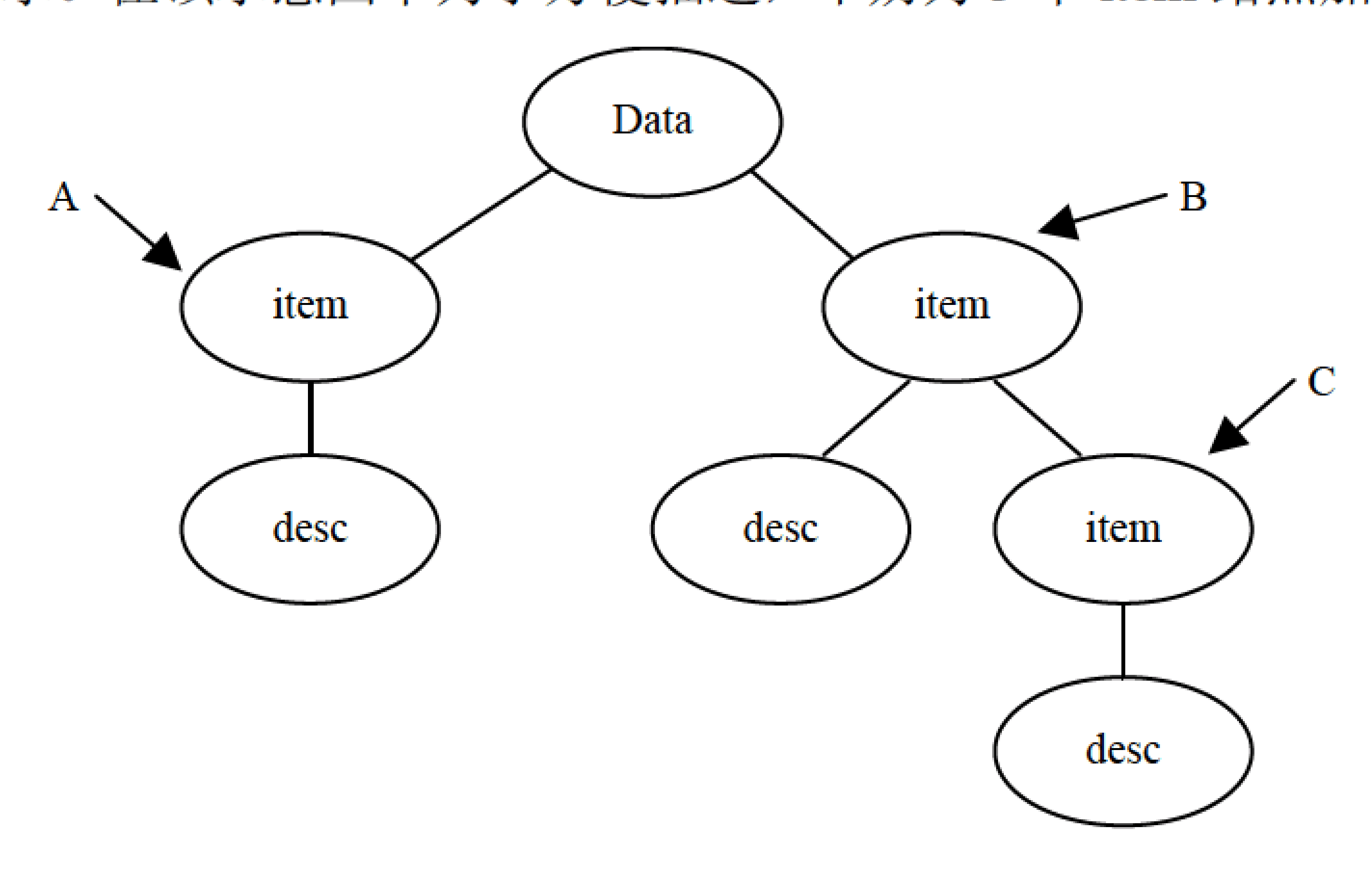


图 9-5　XML 文档的结点树示意图

以下是一些查找路径的表达样例，其中 root=xmlobj.documentElement，也即 root 为 XML 文档的根结点。

- ./item：在目前结点下寻找所有 item 结点。例如：

```
root.selectNodes("./item ")
```

下面脚本与上述脚本作用等价。

```
root.selectNodes("item")
```

结果为 A、B 两结点。

- ❑ //item：在 XML 文档中寻找所有 item 结点。例如：

```
root.selectNodes("//item")
```

结果为 A、B、C 3 个结点。

- ❑ item/desc：找出 item 结点下的 desc 子结点。例如：

```
root.selectNodes("item/desc")
```

结果为 A、B 下的 desc 子结点，不包括 C 下的 desc 子结点。

- ❑ item[@id="1"]：寻找所有 id 属性值为 1 的 item 结点，其中，符号@用来表示后面的名称为一个属性。例如：

```
root.selectNodes("item[@id= '1']")
```

结果为 A 结点。

这里，方括号用来限制满足条件的特定元素。以下是几个典型实例。

- ◆ 选择元素 Data 中的第一个 item 结点：/Data/item[1]
- ◆ 选择元素 Data 中的最后一个 item 结点：/Data/item[last()]
- ◆ 选择元素 Data 中具有 desc 元素的 item 结点：/Data/item[desc]

- ❑ item[desc="项目一"]：寻找子结点 desc 文本内容为“项目一”的 item 结点。例如：

```
root.selectNodes("item[desc= '项目一']")
```

结果为 A 结点。

- ❑ Data//desc：找出 Data 结点下不限多少层次结构下的 desc 结点。例如：

```
root.selectNodes("Data//desc")
```

结果为 A、B、C 下的 desc 子结点。

- ❑ Data/*/desc：找出 Data 结点的孙子辈的 desc 结点。其中，通配符“*”表示可匹配所有结点。例如：

```
root.selectNodes("Data/*/desc")
```

结果为 C 下的 desc 子结点。

- ❑ */desc：找出所有 desc 子结点。例如：

```
root.selectNodes("*/desc")
```

结果为 A、B 下的 desc 子结点。

- ❑ item[@id and desc]：找出含 id 属性和 desc 子结点的 item 元素。例如：

```
root.selectNodes("item[@id and desc]")
```

结果为 A、B 结点。

- Data[item]/desc：找出包含 item 子结点的 Data 结点下的 desc 子结点。例如：

```
root.selectNodes("Data[item]/desc ")
```

　　结果为空（null）。
- //desc[.="项目一"]：找出所有层次下结点内容为“项目一”的 desc 结点。此时“.”代表当前处理的结点内容。例如：

```
root.selectNodes("//desc[.='项目一']")
```

　　结果为 A 结点下的 desc 子结点。

Xpath 提供了很多函数，可以使用 Xpath 函数改进 Xpath 查询，提高 Xpath 的编程能力和灵活性。以下列出部分常用函数的功能。

- count(node_set)：求 node_set 参数中的结点数。
- last()：求结点集中最后一个结点的位置。
- name()：求当前结点的名称。
- position( )：求结点在父级中的索引号。
- contains(string1,string2)：检查第一个参数串中是否包含第二个参数串。
- translate(string1,string2,string3)：返回将 string1 字符串中所有出现的 string2 字符串替换为 string3 字符串后的结果。

在 IE 浏览器端，DOMDocument 对象默认用类似正则匹配的方法来进行结点选取。要使用 Xpath，则必须通过 SelectionLanguage 属性的设置将 Xpath 作为默认的选择语言。

【例 9-5】　在 IE 浏览器端使用 Xpath 函数

```
--------------------------------------------------------ex9-5.htm--------------------------------------------------------
<script type="text/javascript">
var xmldoc=new ActiveXObject('Microsoft.XMLDOM');
xmldoc.setProperty("SelectionLanguage", "XPath");
xmldoc.loadXML("<employees><employee><name>张三</name></employee>
            <employee><name>李四</name></employee></employees>");
var res = xmldoc.selectSingleNode("//employee[last()]");
document.write(res.text);
res = xmldoc.selectSingleNode("//employee[contains(name,'三')]");
document.write("<br>"+res.text);
</script>
```

在浏览器页面输出如下内容：

李四

张三

【应用思考】利用 contains 函数可实现基于关键词的检索，例如，网络课件中反映章节层次的 XML 文件，每个章、节结点均有标题属性，如何实现查找标题中含有搜索关键字的那些结点，并将其标题和内容的 URL 超链接在页面上列表显示出来。

9.3.3　利用 XSL 实现显示

1．XSL 简介

XSL 是专门为 XML 设计的样式语言，主要包括以下两部分。

（1）XSLT：将 XML 源代码转换为另一种格式，此时新的格式可以是 HTML 文档，也可以是新的 XML 文档等。

（2）XSL-FO：提供格式化命令，设定外观样式。

XSL 内容非常丰富，本书只进行概要性介绍，有兴趣的读者可参阅相关书籍。XSL 进行数据内容转换的突出特点是将模式和模板相结合，在 XSL 样式文件中定义一系列模板，XSL 处理器根据 XML 数据的模式寻找匹配的模板，根据模板的转换输出进行数据的变换处理。XSL 的基本工作原理如图 9-6 所示。XSLT 在实施转换时，使用 Xpath 语言来定位 XML 文档的元素。

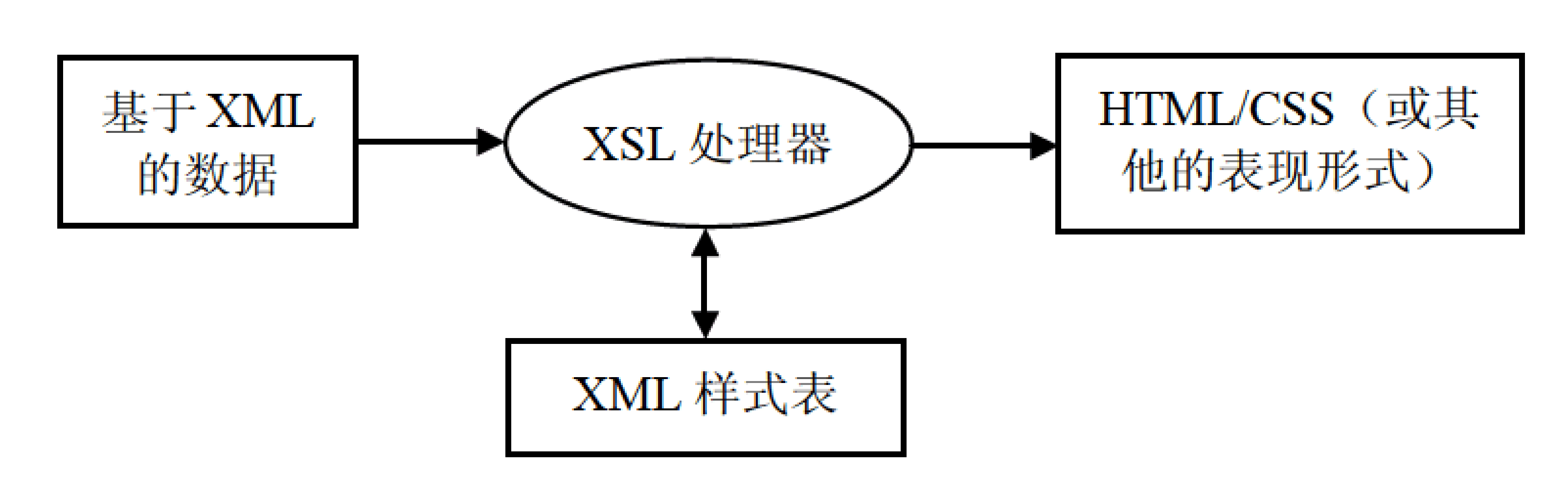


图 9-6　XSL 的基本工作原理

2．应用举例

【例 9-6】　用 XSL 实现 XML 文档显示处理

首先将例 9-4 中 xml 文件的第二行内容改为如下代码：

```
<?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
```

本行表示用 mystyle.xsl 样式文件进行 XML 文档的显示变换。

XSL 样式文件

```
------------------------------------------------------mystyle.xsl------------------------------------------------------
<?xml version="1.0" encoding="GB2312"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<HTML><BODY>
<table border="1"><tr>
<td>姓名</td><td>籍贯</td><td>年龄</td><td>电话</td></tr>
<xsl:for-each select="//student">
<tr><xsl:apply-templates /></tr>
</xsl:for-each>
<xsl:apply-templates select="//student[name='张三']/*"/>
<tr></tr>
</table>
```

```
</BODY></HTML>
</xsl:template>
<xsl:template match="*">
<td><xsl:value-of/></td>
</xsl:template>
</xsl:stylesheet>
```

运行结果如图 9-7 所示。

【说明】本例定义了两个模板，一个是用于匹配根结点的（/）模板，另一个是可匹配所有结点（*）的模板。模板匹配处理的顺序是首先从 XML 文档的根结点出发，寻找匹配模板，根据根结点模板处理的流程顺序处理整个文档，遇到 apply-templates 处理语句再查找相应的匹配模板进行处理。可以看到，针对同一文档可反复引用模板。

程序中，<xsl:for-each select="//student">语句告诉 XSLT 解析器，对于每个 student 结点都必须执行一段变换输出程序，即<tr><xsl:apply-templates /></tr>。

而这里的<xsl:apply-templates />将对每个 student 结点的元素结点应用后面定义的一个模板输出，该模板定义中的“*”代表可匹配所有元素。

```
<xsl:template match="*">
<td><xsl:value-of/></td>
</xsl:template>
```

在上述代码中，<xsl:value-of/>的含义是输出当前结点的文本数据。

在 for-each 循环之后的 apply-templates 语句就是对 name 为“张三”的 student 的所有元素（*），应用模板进行匹配处理。如果将该行写成如下形式，则表示只对 student 元素结点进行模板匹配。

```
<xsl:apply-templates select="//student[name='张三'] "/>
```

则输出结果变为如图 9-8 所示。

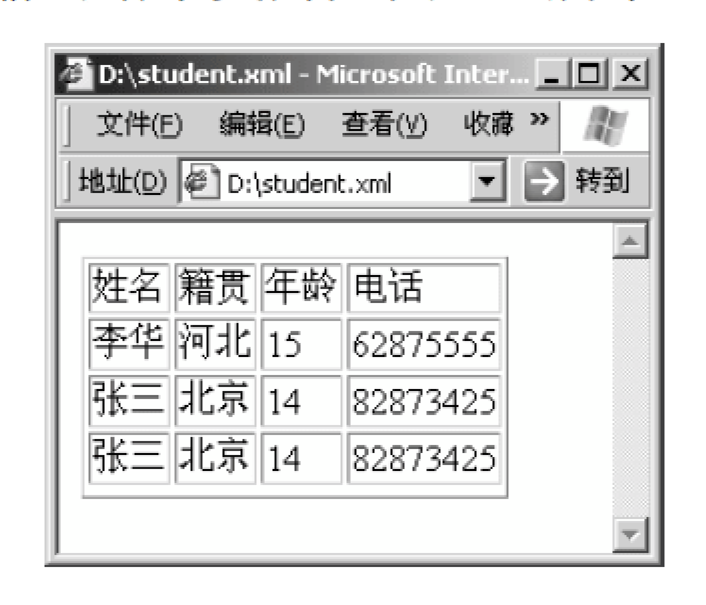


图 9-7　XSL 变换结果显示

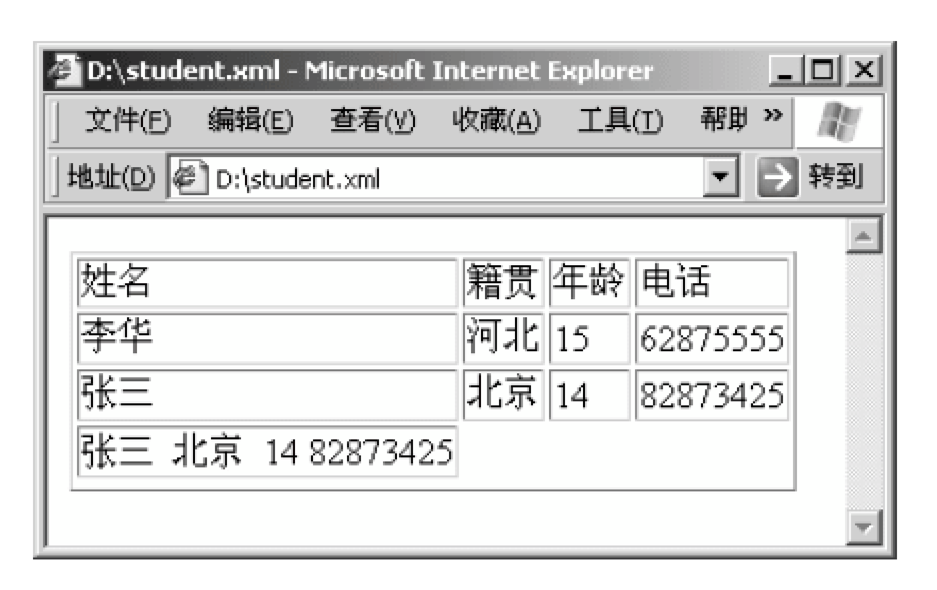


图 9-8　变换后的结果

3．XSL 的优点

XSL 是功能丰富的样式转换描述语言，更多的内容可参考相关书籍。XSL 的主要优点如下：

（1）可在客户端或服务器端转换，XMLDOM 中介绍的 transformNode(stylesheet)方法就是针对某个结点的 XML 子树的内容应用某个 XSL 样式定义进行数据内容的变换处理。

（2）能处理元素、属性、内容。
（3）能为输出添加内容。
（4）能排序或过滤后输出。
（5）支持条件、循环处理。
（6）可使用自定义函数。

9.4　在服务器端访问和处理 XML 文档

XML 广泛用于数据交换处理，XML 数据在某种程度上与数据库是对应的。在实际应用中也可以利用 XML 实现数据的管理，以下结合一个简单的应用样例介绍利用 ASP 对 XML 数据进行增、删、改、查的具体处理方法。

【例 9-7】　利用 XML 实现班级学生管理

本例采用 XML 存储数据，读者可通过本例体会在服务器端对 XML 数据进行增、删、改、查的操作。

文件 1：XML 数据存储格式

```
-------------------------------------------------------class_students.xml-------------------------------------------------------
<?xml version="1.0" encoding="gb2312"?>
<classes>
      <class name="计算机 04-1">
            <student name="张三" sex="男" />
            <student name="张微" sex="女" />
      </class>
      <class name="计算机 04-2">
            <student name="王五 3" sex="男" />
            <student name="李四" sex="女" />
      </class>
</classes>
```

1．数据查询功能——查询某班学生名单

功能要求：通过下拉列表框选择班级，将列出该班所有学生的名单。

文件 2：search.asp 程序

```
----------------------------------------------------------search.asp----------------------------------------------------------
<form method="post" action="search.asp">
<% set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
    xmlDoc.load(Server.MapPath("class_students.xml"))
    set allclass= xmlDoc.selectNodes("//class")
%>
<select   name="classname" size="1" >
   <% for each bj in allclass
        bjname=bj.getAttribute("name")
    %>
    <option value="<%=bjname%>" <%if request("classname")=bjname then%>
```

```
                selected<%end if%>><%=bjname%></option>
     <%next%>
</select>
<input type=submit name="search" value="查询">
</form>

<!-- 以下根据选择的班级查找出本班学生并显示 -->
<% if request("classname")<>"" then
     set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
     xmlDoc.load(Server.MapPath("class_students.xml"))
     set sts= xmlDoc.selectSingleNode("//class[@name='"
            &request("classname")&"']").childNodes
  %>
<table width="80%" border=1 >
<tr><td>姓名</td><td>性别</td></tr>
<% for each student in sts%>
<tr><td><%=student.getAttribute("name")%></td>
<td><%=student.getAttribute("sex")%></td></tr>
<%next%>
</table>
<%end if%>
```

运行结果如图 9-9 所示。



图 9-9　查询班级学生

【说明】这是一个自提交的应用，在表单中通过一个下拉列表框列出所有班级供选择，后面的一段代码通过条件限制只在选择了班级并提交表单后才会执行，班级和班级学生均从同一 XML 文件中查询得到。仔细体会 Xpath 的表示形式及 selectNodes 方法和 selectSingleNode 方法的选择。

2．数据添加——在某班增加一个学生

功能要求：在数据录入表单提供选择班级、输入姓名、性别等输入控件，表单提交后将数据写入 XML 文档。

文件 3：inputstudent.asp 程序

```
----------------------------------------------------inputstudent.asp----------------------------------------------------
<form method="post" action="inputstudent.asp">
```

```
<table width="80%" border=1 >
<tr><td>班级</td><td>
<% set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
    xmlDoc.load(Server.MapPath("class_students.xml"))
    set allclass= xmlDoc.selectNodes("//class")
%>
<select   name="classname" size="1" >
  <% for each bj in allclass
       bjname=bj.getAttribute("name")
    %>
    <option value="<%=bjname%>" <%if request("classname")=bjname then%>
          selected<%end if%>><%=bjname%></option>
  <%next%>
</select></td></tr>
<tr><td>姓名</td><td><input type=text size=10 name="studentname"></td></tr>
<tr><td>性别</td><td><input type=text size=10 name="sex"></td></tr>
</table>
<input type=submit name="input" value="  提交  ">
</form>

<!-- 以下将表单数据写入 XML 文档 -->
<%if request("input")<>"" then
    set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
    xmlDoc.load(Server.MapPath("class_students.xml"))
    set sts = xmlDoc.selectSingleNode("//class[@name='"
           &request("classname")&"']")
    set tmpNode = xmlDoc.createNode("element","student","")
    sts.appendChild(tmpNode)
    tmpNode.setAttribute "name",request("studentname")
    tmpNode.setAttribute "sex",request("sex")
    xmldoc.save(Server.MapPath("class_students.xml"))
end if
 %>
```

【说明】本例也是一个自提交的应用，前面部分为一个数据录入表单，后面部分为将表单提交的数据写入 XML 文档。仔细体会如何创建新的学生结点，并加入到本班的班级结点上，以及如何为结点设置属性，最后如何保存 XML 文档到文件中。

3．数据删除处理——删除某班的某个学生

功能要求：提供一个下拉列表框用来选择班级，在表格中列出班级学生，并为每个学生安排一个删除超链接，单击该超链接可将相应学生删除。

文件 4：delstudent.asp 程序

```
----------------------------------------------------delstudent.asp----------------------------------------------------
<form method="post" action="delstudent.asp">
<%set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
    xmlDoc.load(Server.MapPath("class_students.xml"))
    set allclass= xmlDoc.selectNodes("//class")
```

```
%>
<select   name="classname" size="1" >
  <% for each bj in allclass
     bjname=bj.getAttribute("name")
   %>
   <option value="<%=bjname%>" <%if request("classname")=bjname then%>
             selected<%end if%>><%=bjname%></option>
  <%next%>
</select>
<input type=submit name="search" value="查询">
</form>

<!-- 以下根据选择的班级查找出本班学生并显示 -->
<%if request("classname")<>"" then
    set sts= xmlDoc.selectSingleNode("//class[@name='"
               &request("classname")&"']").childNodes
 %>
<table width="80%" border=1 >
<tr><td>姓名</td><td>性别</td><td></td></tr>
<% k=0
for each student in sts%>
<tr><td><%=student.getAttribute("name")%></td>
<td><%=student.getAttribute("sex")%></td>
<td><a href="delprocess.asp?classname=<%=request("classname")%>
               &xh=<%=k%>">删除</a></td>
</tr>
<%k=k+1
next%>
</table>
<%end if%>
```

【说明】本程序与前面的查询程序的代码基本相同，只是每个学生后跟有一个删除该学生的超链接。注意为超链接传递参数，以便删除程序进行处理。

文件 5：delprocess.asp 程序

```
-----------------------------------------------------delprocess.asp-----------------------------------------------------
<% set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
    xmlDoc.load(Server.MapPath("class_students.xml"))
    set bj= xmlDoc.selectSingleNode("//class[@name='"&
          request("classname")&"']")
    set students=bj.childNodes
    k=cint(request("xh"))
    set student=students.item(k)
    bj.removeChild(student)
    xmlDoc.save(Server.MapPath("class_students.xml"))
    response.redirect("delstudent.asp?classname="&request("classname"))
%>
```

【说明】根据超链接传递的班级和学生参数将该班的某个学生结点从 XML 文档中删

除，注意，学生参数传递的是学生在班级内的结点序号，仔细体会如何从集合中按序号访问一个元素结点。最后，将 XML 文档保存到文件中，网页重定向到本班学生删除显示页面。

4．数据修改处理——修改某个学生的信息

功能要求：选择班级，列出班级的学生，并对每个学生提供一个修改超链接，单击超链接进入该学生信息修改页面，在修改页面将提供一个数据编辑表单，表单中将列出学生的原来信息，学生对信息修改后单击“提交”按钮，将在 modify.asp 程序中根据修改信息更新 XML 文件。

文件 6：search2.asp 程序

```
-----------------------------------------------------search2.asp------------------------------------------------------
<form method="post" action="modify_student.asp">
<%
set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
xmlDoc.load(Server.MapPath("class_students.xml"))
set allclass= xmlDoc.selectNodes("//class")
%>
<select   name="classname" size="1" >
  <% for each bj in allclass
       bjname=bj.getAttribute("name")
   %>
   <option value="<%=bjname%>" <%if request("classname")=bjname then%>
          selected<%end if%>><%=bjname%></option>
  <%next%>
</select>
<input type=submit name="search" value="查询">
</form>
<!-- 以下根据选择的班级查找出本班学生并显示 -->
<%if request("classname")<>"" then
   set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
   xmlDoc.load(Server.MapPath("class_students.xml"))
   set sts= xmlDoc.selectSingleNode("//class[@name='"
          &request("classname")&"']").childNodes
 %>
<table width="80%" border=1 >
<tr><td>姓名</td><td>性别</td><td></td></tr>
<% k=0
for each student in sts%>
<tr><td><%=student.getAttribute("name")%></td>
<td><%=student.getAttribute("sex")%></td>
<td><a href="modify.asp?classname=<%=request("classname")%>
          &xh=<%=k%>">修改</a></td>
</tr>
<%k=k+1
next%>
</table>
<%end if%>
```

【说明】本例与前面的班级学生查询程序基本相同，只是在显示一个学生时提供一个修改该学生信息的超链接。

文件 7：modify.asp 程序

```
--------------------------------------------------------modify.asp--------------------------------------------------------
<% set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
     xmlDoc.load(Server.MapPath("class_students.xml"))
     set bj= xmlDoc.selectSingleNode("//class[@name='"&
          request("classname")&"']")
     set students=bj.childNodes
     k=cint(request("xh"))
     set student=students.item(k)
     name=student.getAttribute("name")
     sex=student.getAttribute("sex")
     age=student.getAttribute("age")
%>
<!-- 以下将学生信息在表单输入框中显示 -->
<form method="post" action="modiwrite.asp">
<table width="80%" border=1 >
<tr><td>班级</td><td><select   name="classname" size="1" >
     <% for each bj in allclass
       bjname=bj.getAttribute("name")
     %>
     <option value="<%=bjname%>"
            <%if request("classname")=bjname then%> selected
            <%end if%>><%=bjname%></option>
   <%next%>
</select></td></tr>
<tr><td>姓名</td><td>
<input type=text size=10 name="studentname" value=<%=name%>></td></tr>
<tr><td>性别</td><td>
<input type=text size=10 name="sex" value=<%=sex%>></td></tr>
<tr><td>年龄</td><td>
<input type=text size=10 name="age" value=<%=age%>></td></tr>
</table>
<input type=hidden name="xh" value=<%=k%>>
<input type=submit name="input" value="  提交  ">
</form>
```

【说明】根据 URL 参数传递的班级和学生序号访问 XML 文档，从班级中得到指定的学生结点，读取结点的相关属性，在修改表单中显示学生的数据。

文件 8：modiwrite.asp 程序

```
------------------------------------------------------modiwrite.asp------------------------------------------------------
<% set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
     xmlDoc.load(Server.MapPath("class_students.xml"))
     set bj= xmlDoc.selectSingleNode("//class[@name='"&
          request("classname")&"']")
```

```
    set students=bj.childNodes
    k=cint(request("xh"))
    set student=students.item(k)
    student.setAttribute "name",request("studentname")
    student.setAttribute "sex",request("sex")
    student.setAttribute "age",request("age")
    xmlDoc.save(Server.MapPath("class_students.xml"))
    response.redirect("modify_student.asp?classname="&request("classname"))
%>
```

【说明】根据提交的学生信息修改 XML 文档的内容，处理完毕后重定向到修改选择页面。

9.5　在客户端访问和处理 XML 文档

9.5.1　通过脚本装载和处理 XML 文档

可以使用 JavaScript 语言来显示 XML 数据。

【例 9-8】　利用 JavaScript 访问 XML 文档实现交互做题

```
---------------------------------------------------doexercise.htm---------------------------------------------------
<html> <head>
<script type="text/javascript">
var pos=0;                                  //当前所做试题的序号
function init() {
  var xmldoc = new ActiveXObject("Microsoft.XMLDOM")
  xmldoc.async="false"                      //关闭异步载入
  xmldoc.load("exercise.xml")               //载入名字为 exercise.xml 的 XML 文档
  qNodes = xmldoc.selectNodes("//question[point='1-0-0']");
  //选取当前学习知识点的试题，假设当前学习知识点为 1-0-0
  total= qNodes.length;                     //求试题数量
  display();                                //显示第 1 道试题的序号和试题内容
}
function check(x) {                         //检查用户解答的正确性
  answer=qNodes.item(pos).childNodes. selectSingleNode("answer").text;
  if (x.value== answer){
    alert(" 对 ! ");
  } else {
    alert(" 错 !");
  }
}
function move(f) {                          //移动试题，根据 f 的值改变试题序号 pos 值
  if (f==0)
    pos=0;
  if (f==2)
    pos=total-1;
```

```
   if ((f==-1)&&(pos > 0))
      pos=pos-1;
   if ( (f==1)&&(pos<total-1))
      pos=pos+1;
}
function display() {   //显示当前试题的序号和内容
   document.all.number.innerHTML="<font color=blue>共有"+total+"题</font>，第"+(pos+1)+"题";
   content.innerHTML="<pre>"+ qNodes.item(pos).childNodes.item(1).text+"</pre>"; //显示试题内容
 }
</script>
</head>
<body onload="init()" > <center>
<!-- 以下定义显示试题序号和试题内容的层 -->
<div id="number" ></div>
<div id="content" align=left></div>
<!-- 以下为试题解答和翻动试题表单 -->
<form>
<table width="60%" border ="0">
<tr>
<td align=center>A
<input type="radio" name="ans" value="A" onclick="check(this)"></td>
<td align=center>B
<input type="radio" name="ans" value="B" onclick="check(this)">
</td>
<td align=center>C
<input type="radio" name="ans" value="C" onclick="check(this)"></td>
<td align=center>D
<input type="radio" name="ans" value="D" onclick="check(this)">
</td>
<td align=center>E
<input type="radio" name="ans" value="E" onclick="check(this)"></td>
</table> <p>
<input type="reset" value="第一题" onclick="move(0)" />
<input type="reset" value="上一题" onclick="move(-1)" />
<input type="reset" value="下一题" onclick="move(1)" />
<input type="reset" value="最后题" onclick="move(2)" />
</form>
</body>
</html>
```

如图 9-10 所示为网络教学系统中单选练习的界面。

【说明】XML 文档的格式见例 9-3。本例编写了如下几个 JavaScript 函数：

- display 函数负责当前序号的试题的显示处理。
- move 函数根据参数值更改当前试题序号，实现试题的翻动处理。
- check 函数根据用户的解答选择与标准答案进行比较，给出对错提示。
- init 函数用于完成初始化处理，包括加载试题的 XML 文档、选取当前知识点的试题和调用 display 方法显示第 1 道试题。

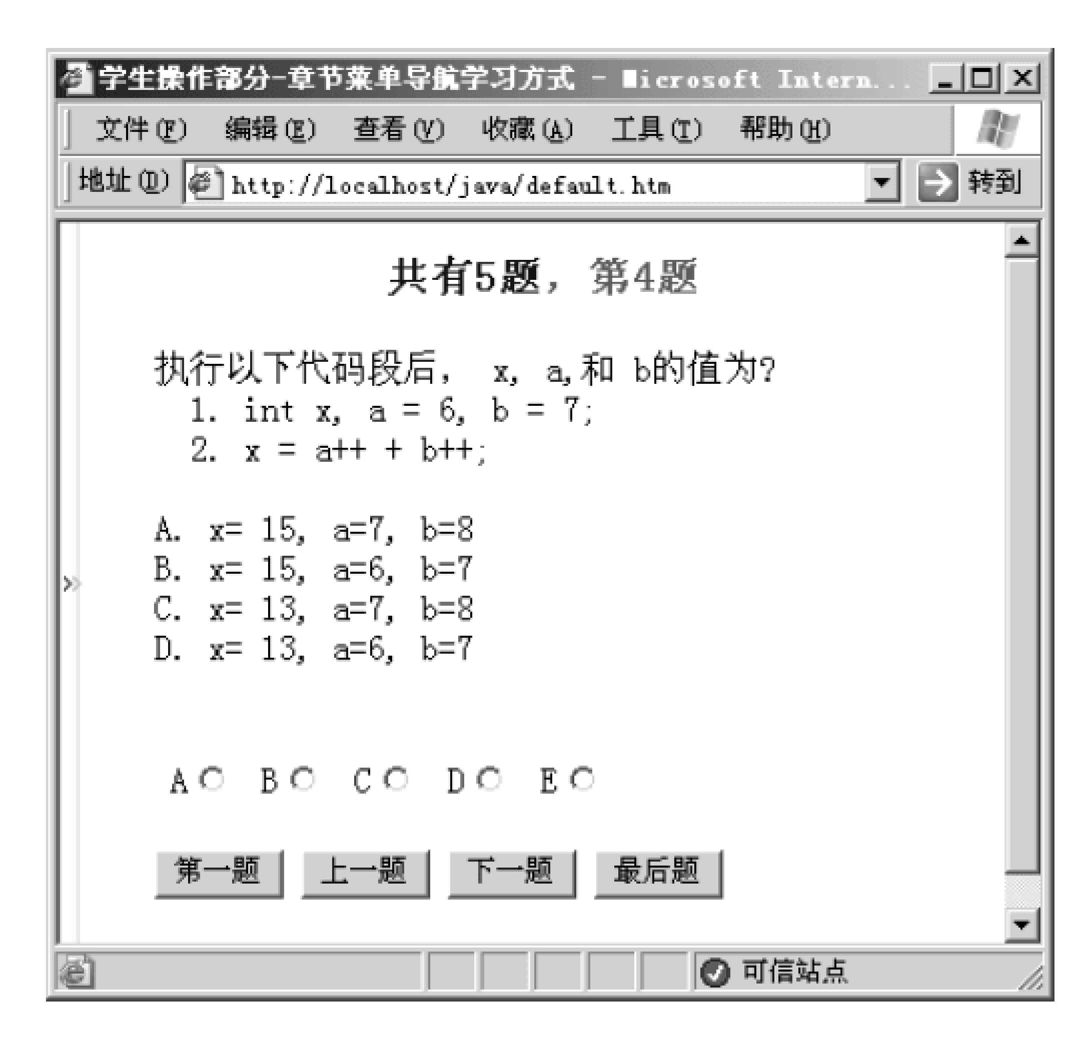


图 9-10　单选练习界面

显示处理采用层定义实现，包括显示试题序号的层 number 和显示试题内容的层 content，试题解答的选项按钮和翻动试题的按钮通过定义 onclick 事件处理方法实现相应的处理。此时在 check 和 move 函数设计中采用了两种典型的参数传递处理，前者采用传递事件源对象（this），后者传递代表性数值。读者可考虑对程序做些改动，修改参数传递设计，分别在函数定义处和调用处做修改。

9.5.2　XML 数据岛

1．XML 数据岛的定义方式

XML 数据岛就是指将 XML 文档加入到 HTML 页面文件中，它以<XML>标记开头，在 HTML 页面文档中通过 ID 属性给定的名称引用数据岛。其具体定义方式有以下两种：

（1）将整个 XML 数据岛直接写入到 HTML 文件中。

例如，记录用户电话号码的 XML 数据岛。

```
<XML ID="xmldso">
<?xml version="1.0" encoding="gb2312"?>
<user_phone>
  <user>
    <name>张三</name>
    <telphone>59313</telphone>
  </user>
  <user>
    <name>李四</name>
    <telphone>59323</telphone>
  </user>
  <!-- 其他 user…… -->
</user_phone>
</XML>
```

（2）外部联接方式，将以上内容保存到 customer.xml 文件中，然后通过数据岛定义语句中的 SRC 属性引用一个外部的 XML 文件。例如：

```
<XML ID="xmldso"   SRC="customer.xml"> </XML>
```

甚至可以通过访问数据库动态产生 XML 数据源。当然，产生的数据格式要符合 XML 要求。

```
<XML ID="xmldso"   SRC="getdata.asp"> </XML>
```

2．利用文档对象模型访问数据岛数据

XML 文档在结构上为树型层次结构，利用文档对象模型（DOM）可访问 XML 中的数据。要得到数据岛的根结点有两种方法：一种方法是通过 XMLDocument 属性；另一种方法是通过 documentElement 属性。例如，以下两条语句均可得到根结点对应的文档内容。

```
document.all("xmlid").XMLDocument.nodeValue;
xmlid.documentElement.text;
```

从根结点出发，借助 DHTML 技术可动态地通过 DOM 模型访问 XML 数据岛的数据，实现网页中的交互性要求。

【例 9-9】　搜索与关键字匹配的教学知识点

以下 XML 数据岛中包含有教学章节目录，每个 node 结点代表章节目录树中的一个知识点，知识点之间的包含关系根据 node 结点的包含层次进行划分。每个结点有 title 属性表示结点标题，url 属性表示描述结点内容的 URL 地址。在进行关键词查找时，不区分大小写。

```
<XML ID="xmldso">
<?xml version="1.0" encoding="gb2312"?>
<directory>
  <node title="Java 语言概述" url="text1-0-0.htm">
    <node title="Java 开发和运行环境" url="text1-1-0.htm"/>
    <node title="Java 语言的特点" url="text1-2-0.htm"/>
      ……
  </node>
  <node title="类与对象" url="text2-0-0.htm">
      ……
  </node>
</directory>
</xml>
<script type="text/javascript">
function process() {
var disp="";                                    //用于拼接要显示的内容
var x = point.value;                            //获取输入关键词
var root = xmldso.documentElement;              //XML 数据岛的根结点
if (x=="") { alert("请输入搜索知识点");return false;}
var   mynodes = root.selectNodes("//node");
if (mynodes.length==0)
```

```
  disp=disp+"无相关知识";
else {
  disp=disp+"<ul>";
  var c=0;
  for (var k=0; k<mynodes.length; k++) {  //循环检查所有结点
    var mytitle = mynodes.item(k).getAttribute("title");
    var t1 = mytitle.toLowerCase();
    var t2 = x.toLowerCase();                    //将数据转化为小写
    if (t1.indexOf(t2)>=0) {                     //查找标题中是否含有要找的关键词
     c=c+1;
     var myurl = mynodes.item(k).getAttribute("url");
     disp = disp+"<li style='color: #FF0000'><a href='"+myurl+"'>"+mytitle+"</a></li>";
   }
}
if (c==0)   disp = disp+"无相关知识";
disp=disp+"</ul>";
}
res.innerHTML=disp;
return false;
}
</script>
</head>
<body>
<P ALIGN=CENTER>
<table width="35%">
<td align=middle>    <!--以下为输入知识点的文本框-->
<input name="point" id="point" type=text size=20>
</td><td align=left>
<INPUT   type=button onclick="process()" name="s1" value=搜索>
</td>
</table>
</P> <div id="res" ></div>
</body>
</html>
```

图 9-11 所示为网络教学系统中的知识搜索界面。



图 9-11　网络教学系统中的知识搜索界面

【说明】页面中嵌入了一个 JavaScript 函数 process()，其功能是将用户输入的关键词与 XML 数据岛中各结点的 title 属性值进行比较，如果能查询到，则在页面的 res 标识位置显示匹配结点的 URL 超链接，否则显示“无相关知识”。

3．数据岛数据与 HTML 文档的捆绑

XML 数据岛中的数据可以通过与特定 HTML 元素的捆绑实现数据显示。

（1）用表格进行数据捆绑

以前面的 XML 文档为例，表格的列对应 user 的各个域，而行则对应每个 user 的数据记录。通过表格的 datasrc 属性指定 XML 数据源对象，在表格栏中通过<span>元素的 datafld 属性捆绑 user 的数据域。例如：

```
<XML ID="xmldso"   SRC="customer.xml"> </XML>
<center>
<table datasrc="#xmldso" border="1">
<tr align="left" >
<td><span datafld="name"></span></td>
<td><span datafld="telphone"></span></td>
</tr>
</table>
</center>
```

执行以上代码后将会在表格中显示所有 user 的数据。运行结果如图 9-12 所示。

根据应用需要，可以实现翻页显示，通过<TABLE>标记中的 DATAPAGESIZE 属性可指定每页的大小，即每页显示的数据行数：

```
<table id="my" datasrc="#xmldso" datapagesize="5" >
```

并在页面中安排实现翻页功能的按钮控件：

```
<input type=Button VALUE="上一页"    onclick="my.previousPage()">
<input type=Button VALUE="下一页"   onclick="my.nextPage()">
<input type=Button VALUE="第一页"   onclick="my.firstPage()">
<input type=Button VALUE="最后一页"   onclick="my.lastPage()">
```

对于含有主子表格关系的 XML 数据岛的数据，有时需要对记录集数据按主子关联显示。例如，以下 XML 数据岛表示了一个学校的若干班，每个班有若干学生。

```
<XML ID="XMLID">
<school>
    <class>
          <name>计算机 99-1</name>
          <student>
               <name>张三</name>
               <sex>男</sex>
         </student>
         <student>
```

```
                <name>李四</name>
                <sex>男</sex>
            </student>
        ……
    </class>
    <class>
        <name>计算机 99-2</name>
    ……
     </class>
</school>
</XML>
```

为了实现如图 9-13 所示的主子关联显示，需要表格套表格。外层表格的 datasrc 为数据岛标识，内层表格的 datasrc 也为数据岛标识，但要在 DATAFLD 属性中指定子记录集对应的数据域。此时显示班级学生的子表中通过表格的 DATAFLD 捆绑 student 数据域。

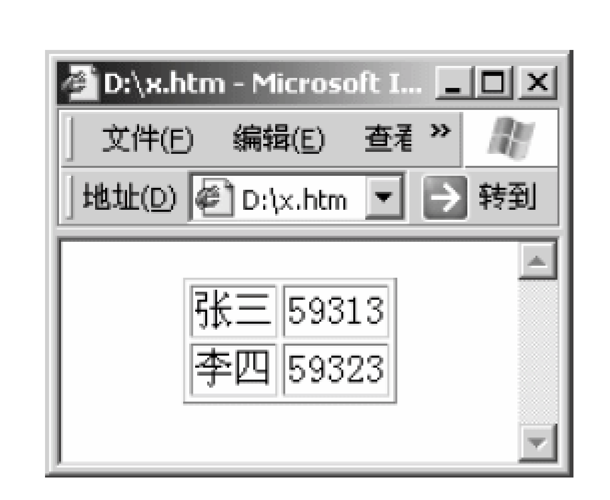


图 9-12　用表格绑定 XML 数据岛



图 9-13　主子关联显示

程序代码如下：

```
<table border="1" datasrc="#XMLID" >
<td><span datafld="name"></span></td>
<td>
<table border="1" datasrc="#XMLID" datafld="student">
<tr><td>
<span datafld="name"></span>
</td><td>
<span datafld="sex"></span>
</td>
</table>
</table>
```

（2）用<span>或<div>标记绑定单个记录的数据

如果不使用表格来显示 XML 数据，则可以在 HTML 文件中自由安排 XML 数据岛数据的显示位置。通过页面中<span>或<div>元素的 datasrc 属性指定要捆绑的数据岛，并通过 datafld 属性指定数据域。但必须注意的是，这种方式一次只能显示一个子结点的数据，一个子结点对应整个数据岛记录集的一个记录，借助数据岛的记录集对象实现对数据岛数据的遍历访问。通过 JavaScript 脚本调用数据岛记录集对象的 movenext()和 moveprevious()等方法可实现记录指针的前后翻动，形式上类似 ASP 的 ADO 记录集的访问。实际上，这里

的记录集的属性和方法与 ADO 中记录集的属性和方法是一致的，即含有 EOF、PageSize、RecordCount、PageCount 等属性，同时也含有 addNew、delete、move、find 等方法。

【例 9-10】 通过数据岛记录集指针的前后移动查看记录

```
---------------------------------------------------- ex9-10.htm -------------------------------------------------------------------
<xml src="customer.xml" id="xmldso" async="false"></xml>
<html> <head>
<script type="text/javascript">
function movenext() {
   x=xmldso.recordset
   if (x.absoluteposition < x.RecordCount){
      x.movenext()
   }
}
function moveprevious() {
   x = xmldso.recordset       //访问记录集
   if (x.absoluteposition > 1) {
      x.moveprevious()
   }
}
</script>
</head>
<body>
<p> 用户名:
<span datasrc="#xmldso" datafld="name"></span>
<br /> 电话 :
<span datasrc="#xmldso" datafld="telphone"></span>
</p> <p>
<input type="button" value="前一用户" onclick="moveprevious()" />
<input type="button" value="后一用户" onclick="movenext()" />
</p> </body>
</html>
```

其中，XML 标记中的 async="false"属性确保 XML 数据在 HTML 文件处理前装入。

数据岛是 IE5.0 以后加入的新特性。由于 XML 数据岛将数据带到了客户方，这就为需要在客户方进行数据处理的应用提供了很大的方便，同时使应用的效率也得到了提高。

本 章 小 结

本章结合具体应用介绍了 XML 应用的主要方面和技术处理，主要内容包括 XML 文档的基本格式、XML 文档的显示处理技术、在服务器端和客户端对 XML 文件进行访问和处理的技术以及对 XML 数据岛的使用等。本章内容实用性较强，重点在于 XML 文档的访问处理，其中一些样例来自作者在 Web 应用开发中的经验总结。

习　　题

1．选择题

（1）XML 文档的根结点在 XSL 中使用（　　）来代表。

A．“*”　　B．“/”　　C．“@”　　D．“?”

（2）XSL 样式单中用到了许多具有一定功能的 XSL 元素和指令，（　　）元素可以定义模板规则，（　　）指令可以应用匹配的模板规则，（　　）指令可以取得特定的结点或表达的值。

A．<xsl:value-of>　　B．<xsl:template>

C．<xsl:apply-template>　　D．<xsl:sort>

（3）XSL 中要匹配任意名称的元素结点，应使用（　　）符号。

A．“*”　　B．“/*”　　C．“/”　　D．“.”

（4）XML 的数据存取机制包括（　　）。

A．可以通过 DOM（Document Object Model）读取 XML 文档中的结点

B．通过 DSO（Data Source Object）进行 XML 的数据绑定可以方便地将 XML 结点同 HTML 标记捆绑，从 XML 文档中读取或写入数据

C．样式单 CSS 和 XSL 实际上可通过为 XML 数据赋予一定的样式信息来使其能够在浏览器中显示

D．XML 直接输出文档

（5）一个 XML 文件最基本的构成是（　　）。

A．XML 声明　B．处理指示　　C．XML 元素　　D．文件的内容

（6）一个 XML 文件能被浏览器解析的最小要求是（　　）。

A．这个文件是形式良好的

B．这个文件是结构完整的

C．每个标记都必须是 XML 标准定义过的

D．扩展名必须是.xml 的文件

（7）使用 XSL 定义 XML 文档的显示方式的基本思想是（　　）。

A．通过定义转换模板，将 XML 源文档转换为带样式信息的可浏览文档

B．定义不同于以往的显示风格

C．控制 XML 文档显示数据的输出

D．通过定义显示模板显示指定的 XML 数据

（8）下列关于 XML 接口 DOM 的描述，错误的是（　　）。

A．DOM 的全称是 Document Object Model，即文档对象模型

B．在应用程序中，基于 DOM 的 XML 分析器将一个 XML 文档转换成一个对象模型的集合（通常称 DOM 树）

C．通过 DOM 接口，应用程序可以在任何时候访问 XML 文档中的任何一部分数据，因此，这种利用 DOM 接口的机制也被称为随机访问机制

D．DOM 强制使用树模型来访问 XML 文档中的信息，不适合 XML 的模式

（9）关于 XSL 模板和模板规则，下列说明错误的是（　　）。

A．XSL 文档包含一组或几组模板规则和其他规则

B．模板规则拥有模式（pattern）以及模板（template）

C．每个模板规则都是 xsl:template 元素

D．模式指定模板规则所使用的树型结构，而模板是用来在与此模式匹配时进行输出的

2．编程题

（1）编程将网上教学数据库中的学生问题转化为 XML 表示形式，利用 XML 数据岛将数据带到客户端，利用 JavaScript 脚本技术读取学生问题进行显示，要求能进行翻页显示。

（2）利用 ASP 加 XML 实现一个简单的个人留言簿，每条留言的内容包括内容、留言时间，要求能分页查看留言。任何人可输入留言，管理员账户可删除留言。

（3）利用 XML 数据岛实现交互做单选题的 Web 页面应用，每道试题含有一个原因解析。要求在同一页面中显示所有试题，当学生做错时，提示做错，做对时弹出消息显示原因。

第 10 章　AJAX 技术

10.1　什么是 AJAX

AJAX 是 Asynchronous JavaScript and XML（异步 JavaScript 和 XML 技术）的简称，通过这种技术，开发人员可以将来自服务器的信息无缝地更新部分 Web 页面或 Web 应用程序，有效弥补用 B/S 方式开发交互式 Web 页面的不足。

AJAX 技术所涉及的主要方面如下。

- XHTML 和 CSS：定义文档的外观和显示样式。
- 文档对象模型（DOM）：访问文挡内容。
- JavaScript 脚本语言：编写执行逻辑。
- XMLHttpRequest 对象：实现与服务器的交互。

其中，XHTML 是可扩展标记语言，它与 HTML 的区别在于 XHTML 遵循严格的 XML 规则，强调标记要配套，嵌套时不能出现交叉。

XMLHttpRequest 对象是 AJAX 技术的核心，其工作原理如图 10-1 所示。XMLHttpRequest 对象是附在 MSXML 链接库中的，它提供了客户端同 HTTP 服务器通信的协议。客户端可以通过 XMLHttpRequest 对象向 HTTP 服务器发送请求并使用 XML 文档对象模型处理回应。如今的绝大多数浏览器都增加了对 XMLHttpRequest 的支持。

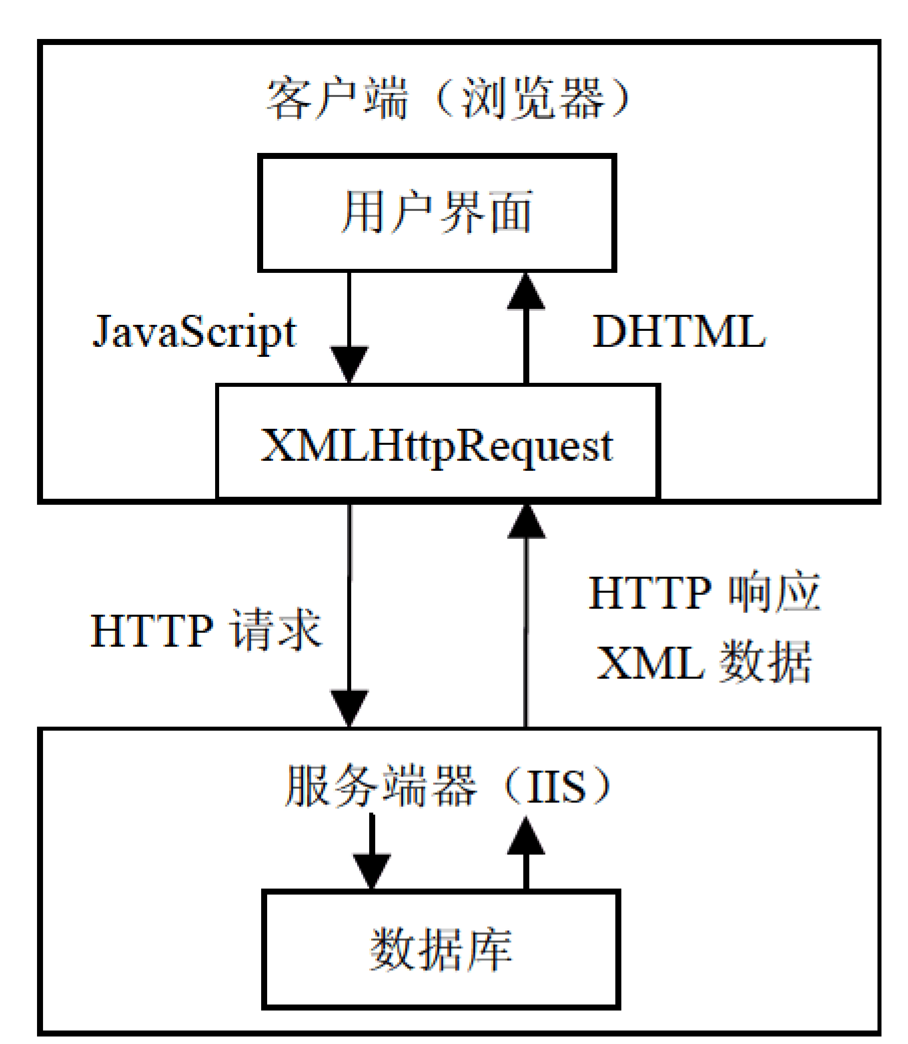


图 10-1　XMLHttpRequest 工作原理

AJAX 利用 JavaScript 事件实现 Web 应用中对用户行为触发的实时响应和处理。

适合使用 AJAX 技术的应用主要有部分页面更新、不可见的数据检索、不间断更新、

平滑的界面、拖放等。而 AJAX 的突出缺点是可能会动态改变 HTML 标签，在一定程度上阻碍了搜索引擎编制索引。

使用 AJAX 的条件是不能禁用脚本语言且需要高版本的浏览器，如 IE4+、MozillaFirefox、Netscape 7+等。

10.2　XMLHttpRequest 对象的使用

10.2.1　创建 XMLHttpRequest 对象

浏览器对 XMLHttpRequest 对象的支持与浏览器的类型和版本有关。创建 XMLHttpRequest 对象时，必须先检测所使用的浏览器是哪种类型。

对于 IE7 以前的 IE 的创建方法是：

```
var xHRObject = new ActiveXObject("microsoft.XMLHTTP");
```

对于 IE7 和其他浏览器的创建方法是：

```
var xHRObject = new XMLHttpRequest( );
```

考虑到兼容不同浏览器的使用，XMLHttpRequest 对象的创建代码可设计如下：

```
var xmlhttp = false;
function creatXmlHttpRequest( ) {
  if (window.XMLHttpRequest) {            //IE7 和其他浏览器的创建方法
      xmlhttp = new XMLHttpRequest();
  }
  else if (window.ActiveXObject) {        //IE4、IE5、IE6 的创建方法
      xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
  }
}
```

【注意】JavaScript 是区分大小写的，代码中 XMLHttpRequest 的大小写要正确。

10.2.2　XMLHttpRequest 对象的属性

XMLHttpRequest 对象具有如下属性。

- onreadystatechange：指定当 readyState 属性改变时的事件处理函数。
- readyState：返回当前请求的状态。返回值意义如表 10-1 所示。

表 10-1　XMLHttpRequest 对象的 readyState 属性的取值

取　值	含　义	解　释
0	未初始化	已建立 XMLHTTP 对象，但尚未调用 open 方法建立到服务器的连接
1	加载中	XMLHTTP 对象已加载，但尚未调用 send 方法发送请求

续表

取　值	含　义	解　释
2	已发送	已调用 send 方法，但服务器尚无响应
3	正在接收	已取得部分服务响应结果，但还不能使用该对象的属性和方法，因为状态和响应头不完整
4	完成	已接收所有返回数据

- responseBody：以无符号字节数组形式返回响应信息。
- responseStream：以 ADO Stream 对象的形式返回响应信息。
- responseText：以字符串形式返回响应信息。
- responseXML：将响应信息格式化为 XML Document 对象表示。
- status：返回当前请求的 http 状态码。
- statusText：返回当前请求的响应行状态信息。

10.2.3　XMLHttpRequest 对象的方法

XMLHttpRequest 对象的方法如表 10-2 所示。

表 10-2　XMLHttpRequest 对象的方法

方　法	功　能
Abort()	取消当前请求
GetAllResponseHeader()	获取响应的 http 头的所有信息
GetResponseHeader()	从响应信息中获取指定的 http 头的信息
open(method, URL, synchronous)	创建一个 http 请求，并指定此请求的方法、URL 以及异步/同步方式。其中，第一个参数 method 有 GET 和 POST 两种方式；第三个参数值为 True 时代表异步，值为 False 时代表同步
Send(String)	发送请求到 http 服务器
setRequestHeader	单独指定请求的某个 http 头

XMLHttpRequest 对象发送请求处理的顺序通常如下：

（1）调用 open 方法，参数中指定服务器 URL 地址等信息。

（2）根据需要设置 onreadystatechange 事件处理函数。

（3）根据需要使用 setRequestHeader 方法设置请求头信息。

（4）使用 send 方法向服务器送出请求。

（5）从 4 个 response 属性中选择一个取得服务器的响应结果。

其中，第（2）步和第（3）步根据需要设定。以下就上述方法中参数的选择做几点说明：

（1）关于 POST 和 GET 方法

POST 方法传递的信息量较大，最多可达 2GB，而 GET 方法传递的信息量较小，IE 限制为 2083 个字符，Opera 为 4050 个字符，Netscape4 为 8192 个字符。

（2）关于查询参数串的传送方式

查询参数串可跟在 URL 串内，也可以通过 send 方法的参数给出。

```
xmlhttp.open("GET", "Reponse.asp&value=1", "true");
xmlhttp.send(null);
```

根据需要还可对查询参数串的内容进行编码，以防止因特殊字符导致 URL 解析错误。

```
var argument = "value="+encodeURIComponent(data);
xmlhttp.open("POST", "Reponse.asp", "true");
xmlhttp.setRequestHeader("Content-Type",
          "application/x-www-form-urlencoded");
xmlhttp.send(argument);
```

其中，encodeURIComponent()是 JavaScript 的一个全局函数，可把字符串作为 URI 组件进行编码。该方法不会对 ASCII 字母和数字进行编码，也不会对诸如-、_、.、!、~、*、'、( )等 ASCII 标点符号进行编码，其他字符（例如，;、/、?、:、@、&、=、+、$、,、# 等是用于分隔 URI 组件的符号）将由十六进制的转义序列替换。

另一个函数 decodeURIComponent()将对 encodeURIComponent()函数编码的 URI 进行解码。

（3）关于同步和异步使用方式

open 方法中的第三个参数用于表示同步或异步方式。在 prototype 等 js 类库中一般默认为异步方式，即设为 True。

同步情况时，js 会等待请求返回，并获取状态信息，不需要 onreadystatechange 事件处理函数。以下为常见编程处理形式：

```
function search( ){
    var URL=" Reponse.asp";
    xmlhttp.open("GET",URL, false);
    xmlhttp.send(null);
    var result = xmlhttp.status;
    if(result==200)  {     //表示 OK ，正常
      alert(xmlhttp.responseText);
    }
}
```

异步情况使用 XMLHttpRequest 对象，须用 onreadystatechange 属性指定事件触发要执行的函数。在事件处理函数中，将检查 readyState 属性，看数据是否准备就绪，如果已经准备就绪，则可读取响应数据。

以下代码在页面加载时通过 XMLHttpRequest 对象的 open 方法装载 student.xml 文件，并通过 XMLHttpRequest 对象的 onreadystatechange 属性定义事件处理回调函数。在回调函数中，第一个要执行的任务是检查数据是否已下载完毕，通过检查 XMLHttpRequest 对象的 readyState 属性是否等于 4（代表已完成下载）来实现。确定数据已下载完毕后，可以用 XMLHttpRequest 对象的 responseText 或 responseXML 属性来取回数据。

```
<SCRIPT LANGUAGE = "JavaScript">
var XmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
function init () {
    XmlHttp.onreadystatechange = doChange;
```

```
        XmlHttp.open("GET", "student.xml", true);
        XmlHttp.send();
}
function doChange() {
        if (XmlHttp.readyState == 4) {
                alert("XML 文件装载完毕");
                alert(XmlHttp.responseText);
        }
}
</script>
<body onload= "init()">……</body>
```

10.2.4　在 Web 服务器端使用 XMLHttpRequest 对象

在 Web 服务器端也可以使用 XMLHttpRequest 对象访问 Internet 上的信息资源。利用 Internet 网上搜索引擎提供的访问接口，通过发送 URL 请求，并对返回的搜索结果进行过滤处理，就可以实现个性化的搜索服务。以下是利用百度搜索服务的 ASP 程序代码：

```
<%
    Set objXMLHTTP = Server.CreateObject("MSXML2.XMLHTTP")
    url="http://www.baidu.com/s?wd="&request("keyword")&
                "&pn="&request("start")    '参数 wd 为关键词，pn 为页码
    objXMLHTTP.open "GET",url, False
    objXMLHTTP.send(null)
    content = objXMLHTTP.responseText
    ……对搜索引擎的结果进行处理，并按自己的形式显示……
%>
```

【注意】有些搜索引擎提供的访问接口进行了限制，如 Google 搜索服务，从服务器上发送 XMLHTTP 请求，则只有在 Google 注册了密钥的站点才能访问，并在访问时通过 URL 参数 key 传递密钥，但当站点为 localhost 时不受限制。

10.3　AJAX 应用举例

10.3.1　样例 1——网络考试中避免并发交卷的处理

在网络考试的交卷处理、文件上传等应用中，涉及的处理任务要花费较多的时间，为了避免服务器处理负担过重而出现问题，可限制 3 秒内不同时处理两个用户的交卷请求。此时可利用 AJAX 技术发送请求到服务器检查是否存在较短时间的“并发”访问，如果存在，则提示用户暂缓交卷。具体操作步骤如下。

首先在表单提交时通过 onsubmit 属性设置事件执行函数，该函数将通过 AJAX 技术发送请求到服务器检查是否存在“并发”，只有当服务器返回的结果为 ok 时才能正常交卷。

```
<form method="post" onsubmit="return check()" action="finish.asp">
```

程序 1：表单提交的执行检查函数 check()

```
function check () {
      xmlhttp.Open("get", "check_exam.asp", false);
      xmlhttp.send(null);
      res=xmlhttp.responseText;
      if (res=="ok")
         return true;
      else{
         alert("服务器忙，请稍后交卷!");
         return false;
    }
}
```

【说明】本例在 open 方法中设置为同步访问方式，也就是客户端脚本在发送请求后将等待服务器的响应，并通过 XMLHttpRequest 对象的 responseText 获取响应信息。

程序 2：实现并发检查的 ASP 程序

服务器实现并发检查的一个方法是用 Application 对象记下前一个用户的访问时刻（用相对秒值表示），若当前用户的访问时刻与前一个用户的访问时刻差在 3 秒内，则禁止访问。

```
---------------------------------------------- check_exam.asp ----------------------------------------------------
<%
response.expires=0
k = now
my=minute(k)*60+second(k)            '计算本用户的访问时刻
if application("examone")<>"" then
  if   abs(my-application("examone"))<=3 then
     response.write "not allow"        '返回不允许访问
     response.end
  else
     application.lock
     application("examone")=my         '记录当前用户的访问时刻
     application.unlock
  end if
else
  application.lock
  application("examone")=my            '记录第一个用户的访问时刻
  application.unlock
end if
response.write "ok"                    '返回允许访问
%>
```

【说明】对于涉及任务处理比较重的应用（如复杂报表的生成），可考虑用此方法限制并发访问用户数，否则服务器的处理程序可能会因为进程负担过重而出现响应上的问题或故障。

10.3.2　样例 2——作品的投票处理

网络作品（如文章、网页等）的发布与评价是目前网络站点的一个常用功能，它为用户之间的协作交流提供了一个很好的空间。在网络教学系统中，允许学生上传作品，并将自己的作品发布出来，用户在浏览作品的同时，可以给作品投票，从而实现学习、交流和评价的有效结合。本例利用 XML 实现信息存储，利用 ASP 技术实现数据访问处理，利用 AJAX 技术实现页面动态处理。用户给作品投票后，可以在不刷新页面的情况下实现投票结果的自动更新显示，能够产生较好的视觉效果，应用效率也较高。

（1）存储评价信息的 XML 文件

以下为存储用户对发表的网页作品进行投票的 XML 文件格式，其中，userpage 结点的 url 为被评网址，name 为用户对作品的命名，username 为作品设计者的用户标识；toupou 子结点记录各用户对网站的投票，byuser 为投票者，amount 为投票星级。

```
---------------------------------------------- homepage.xml ---------------------------------------------------------
<?xml version="1.0" encoding="gb2312"?>
<allpages>
<userpage url="page_1.htm" name="我的个人网页" username="jsj04115">
   <toupou byuser="ding" amount="2"/>
   <toupou byuser="jsj04118" amount="4"/>
   ……
</userpage>
……
</allpages>
```

（2）应用界面设计

应用界面由 ASP 脚本访问 XML 文件和数据库动态产生，用户自己发布的作品还提供有删除超链接，如图 10-2 所示。



图 10-2　网页作品投票界面

界面的关键设计是投票下拉列表框和评价等级的显示处理。在下拉列表框的生成处理中定义了 onchange 事件执行处理程序 vote()，该函数设计有 3 个参数，分别为网站 URL、下拉列表框对象及作品序号。其中，作品序号是为了更新页面的评价等级，各作品的投票评价等级对应的显示区域的标识名按照序号分别为 vote1、vote2…，可通过一个循环实现此设置。以下为生成下拉列表框和投票结果显示的代码段，其中，num 为被评价作品对应的序号。在生成界面时还要考虑在评价等级栏中显示作品的已有投票结果，对于已投票的作品在投票列表框中显示该用户的投票值作为列表框的默认选中项。

```
<%
set  userpage=xmldoc.documentElement.selectNodes("//userpage")
num = 0
for  xh = userpage.length-1 to 0 step -1          '循环处理所有发布的作品
num=num+1                                          '作品排列序号
set node = userpage.item(xh)                       '访问某个作品的结点
m = node.getAttribute("username")                  '发布者标识
url = node.getAttribute("url")                     '发布的网页地址
          ……//其他显示部分略
<td  width="118"  align=center><p  align=center><select  size="1"  name="D<%=num%>"
onchange="vote('<%=url%>',this,<%=num%>)">
<option value="0"> </option>
<option value="5" <%if piao=5 then%> selected<%end if%>>*****</option>
<option value="4" <%if piao=4 then%> selected<%end if%>>****</option>
<option value="3" <%if piao=3 then%> selected<%end if%>>***</option>
<option value="2" <%if piao=2 then%> selected<%end if%>>**</option>
<option value="1" <%if piao=1 then%> selected<%end if%>>*</option>
</select></p>
</td>
<% set votes = node.selectNodes("toupou")
……计算投票综合等级 xin 的代码略，同后面的投票记录程序……
%>
<td width="118" align=center><span id="vote<%=num%>"><%=xin%></span></td>
……
```

（3）投票事件的引发与处理

当用户通过下拉列表框进行投票时将触发 onchange 事件，从而导致执行 vote 函数。该函数的功能是根据用户的投票选择提交信息给服务器的 ASP 程序进行记录处理。这里，利用 XMLHTTP 对象提交 URL 访问请求，并利用该对象的 responseText 属性获取服务器的响应信息，借助 DHTML 将响应信息在相应页面位置进行更新。

```
<SCRIPT LANGUAGE = "JavaScript">
var xmlhttp= new ActiveXObject("Microsoft.XMLHTTP");
function vote(url, obj,n) {
  myurl="updatevote.asp?url="+url+"&choice="+obj.value;
  //将投票选项传递给处理程序，传递 url 参数是为了在处理程序中查找被投票对象
  xmlhttp.Open("POST", myurl, false);
  xmlhttp.send(null);                                   //发送请求
```

```
  x= xmlhttp.responseText;                              //获取服务器的响应
  if (x!="fail")
      document.getElementById("vote"+n).innerHTML=x;    //更新页面评价等级
else
  alert("每个网页只能投一次票");
}
</script>
```

【说明】本例用 getElementById("vote"+n)查找页面上显示投票的标记，方法参数 n 根据实际选择动态变化，这是实现动态显示处理的一个技巧。

（4）投票记录程序

投票记录程序的功能是检查用户的投票是否有效，对任意一个网站的投票，一个用户只能有一次表决权。如果投票有效，则记录该票的信息，并计算相应网站的投票综合表决结果返回客户浏览器；如果投票无效，则返回 fail 消息。

以下为投票处理程序，值得一提的是，程序中用到的 Cookie 变量 username 是用户登录后记录的用户身份标识。

```
------------------------------------------------- updatevote.asp ------------------------------------------------
<% set xmldoc = Server.CreateObject("Microsoft.XMLDOM")
xmldoc.async = false
xmldoc.load(Server.MapPath("homepage.xml"))
set p=xmldoc.selectSingleNode("//userpage[@url='"+request("url")+"']")
'根据 url 参数查找被评结点，每个作品的 URL 是唯一的
set myvote=p.selectSingleNode("toupou[@byuser='"+request.cookies("username")+"']")
'查找是否有当前用户的投票
if not (myvote is nothing) then
  response.write "fail"                                 '已投过票，不能再投
  response.end
end if
set my=xmldoc.createElement("toupou")                   '创建一个新的投票结点
my.setAttribute "byuser" , Request.cookies("username")  '投票用户标识
my.setAttribute "amount" , request("choice")
p.appendChild(my)                                       '作品加入新的投票结点
xmldoc.save(Server.MapPath("homepage.xml"))             '保存修改后的 XML 文件
'以下计算投票等级
set votes=p.selectNodes("toupou")
k=votes.length
xing=""
x=0
if k<>0 then
  for each vote in votes
    x=x+cint(vote.getAttribute("amount"))
  next
  x=(int)(x/k)                                          '计算平均等级分
  for i=1 to x
    xin=xin&"*"                                         '将等级分转化为星级串
  next
end if
```

```
response.write xin                                   '将投票等级返回浏览器
%>
```

10.3.3　样例 3——页面元素间的关联处理

Web 页面中多元素的关联查询在应用设计中比较常见。例如，在学生作业解答的查询页面中，用户选择了某个班，则学生选项应是该班的学生。以往的处理办法是根据用户选择的班级，通过事件处理机制，导致表单提交，从而根据所选班级查询数据库表格将相关学生显示在新的页面中。这样的显示处理缺点主要有 3 点：一是页面重新生成，因页面的刷新导致页面显示的闪烁感；二是处理效率低，重新生成页面要重写页面上所有区域的内容；三是编程比较烦琐，随着页面数据相关性的增加，需要传递和判断的信息量会逐渐增加，给编程带来的困难也随之增加。本例采用 AJAX 技术实现页面元素间的关联处理，只需要根据页面某个元素的变化对其关联的元素进行改变即可，而不用改变页面上的其他内容，从而实现在页面不刷新的情况下实现数据关联的处理，如图 10-3 所示。

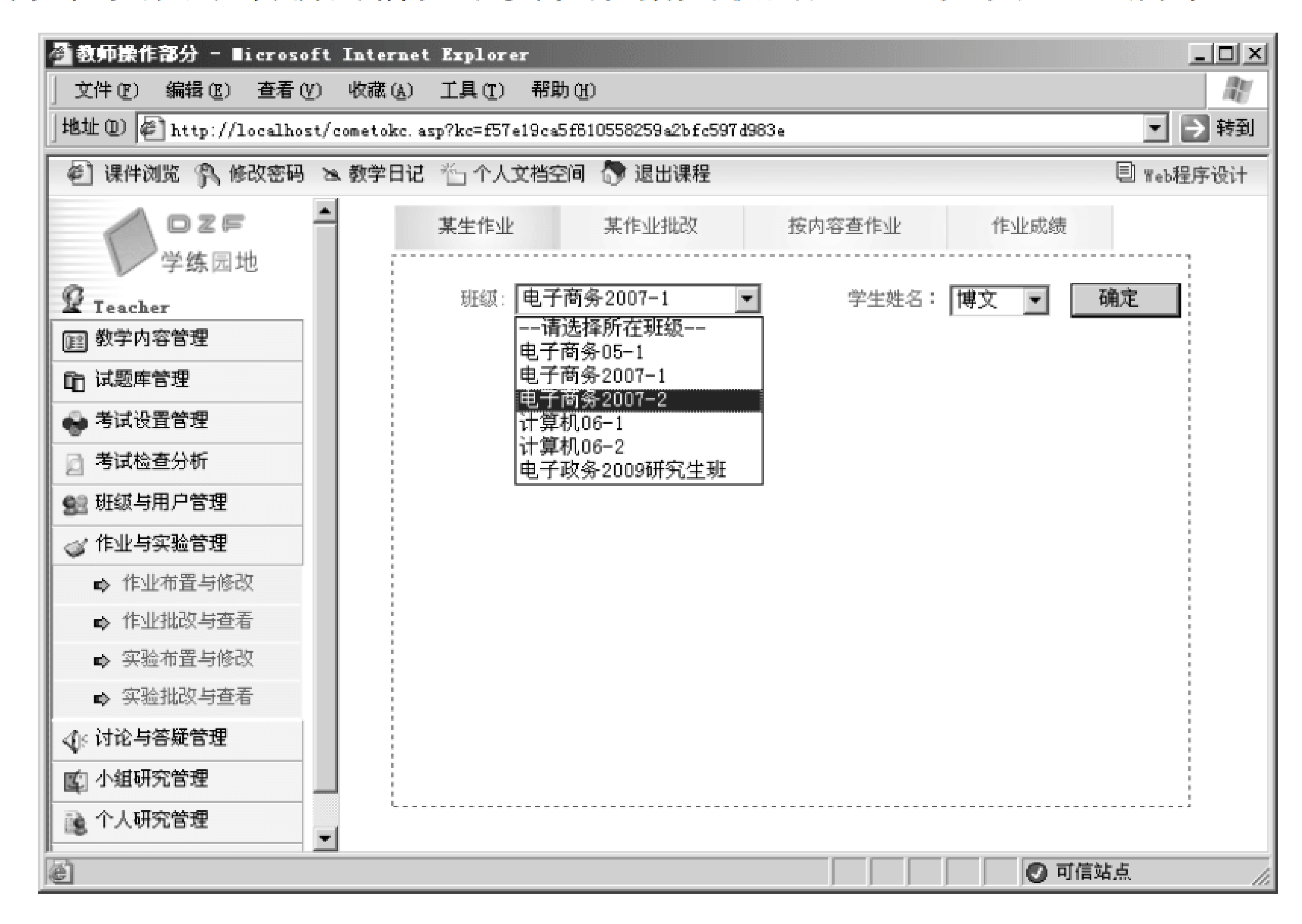


图 10-3　班级与姓名的关联显示处理

在页面中对应“班级”和“学生姓名”的选择框分别命名为 classid 和 studentid，在“班级”对应的下拉列表框定义事件函数 onChange="check()"，check 函数的作用是根据用户选择的学院通过 XMLHTTP 对象访问服务器上的 ASP 程序，并通过读取 ASP 程序的响应信息更新“学生姓名”列表框的显示内容。

1．页面元素关联设计

在本应用示例中，要讨论的两个关联元素为两个下拉列表框，这两个下拉列表框分别用来选择班级和学生。为了实现两个关联元素间的连锁响应，可在代表班级的下拉列表框中定义 onchange 事件处理函数（check）。

```
……
<td width="70" align="right">班级: </td>
<td   height="30">
<select name="classid" size=1 onChange="check( )">
<option value= "" >--请选择所在班级--</option>
<%
sql="select *   from   classtable "      '查班级表
set rs=conn.execute(sql)
do while not rs.eof                      '用循环列出所有班级
%>
 <option value="<%=rs("classid")%>"><%=rs("classname")%></option>
<%rs.movenext
loop
rs.close
%>
</select></td>
<tr bgcolor=#e3f1d1>
<td width="110" align="right">学生姓名：  </td><td   height="30">
<select name="student" size="1">
</select>
</td>
</tr>
……
```

2．事件处理代码

在事件处理函数 check 中，通过 AJAX 技术访问服务器端的 ASP 程序，并将选择的班级作为参数传递给该 ASP 程序。接下来读取 ASP 程序返回的该班级所有学生的信息，返回的信息中各学生之间用逗号隔开，程序中用字符串对象的 split 方法将各学生分离出来存储到一个数组中。然后循环处理数组元素，用 DHTML 技术更新页面上的学生选项。

```
<script language="JavaScript">
var xmlhttp= new ActiveXObject("Microsoft.XMLHTTP");
function check () {
    myurl="find_people.asp?classid="+classid.value;
    xmlhttp.Open("POST", myurl, false);
    xmlhttp.send(null);
    res = xmlhttp.responseText;              //获取 ASP 程序的响应
    x1 = res.split(",");                     //分离出班级学生
    student.options.length=x1.length;
    for (i =0;i<x1.length-1;i++) {
       m = x1[i].split(":");
       student.options(i).text = m[1];       //更新页面上学生下拉列表框的选项
       student.options(i).value = m[0];
    }
}
</script>
```

【说明】下拉列表框中的每个选项均有文本内容（text）和数据值（value）两个属性，

每个学生的数据信息对应这两部分的分别为姓名和登录标识，两者之间用冒号分隔。

3．根据班级访问数据库获取学生的 ASP 程序

该 ASP 程序将从数据库表格中查询所选择班级的学生，将各个学生按逗号分隔的文本串形式返回给请求者，学生的用户标识（user-id）和姓名（username）之间用冒号分隔。其中，包含文件 conn.asp 用于建立与数据库的连接。

```
------------------------------------------------ find_people.asp ------------------------------------------------
<!--#include file="conn.asp"-->
<% response.ContentType= "text/plain"
response.CharSet = "gb2312"
sql= "select *   from usertable where classid='"&request("classid")&"'"
set rs = conn.execute(sql)        '通过连接对象执行 SQL 查询得到结果集
do while not rs.eof
   response.write   rs("user_id")&":"&rs("username")&","
                              '输出的学生之间用逗号隔开
   rs.movenext
loop
%>
```

10.4　在 AJAX 中使用 JSON

向服务器发出 AJAX 请求时，可以以两种不同的方式从服务器响应中检索数据，一种是 responseXML，另一种是 responseText。前者以 XML 格式检索数据，后者以纯文本格式解析数据。在数据传输表示上目前流行 JSON（即 JavaScript 对象表示法），它是一种轻量级的数据交换格式，易于用户阅读和编写，同时也易于机器解析和生成。

JSON 采用完全独立于语言的文本格式，它是基于 JavaScript 的一个子集。

10.4.1　JSON 的具体形式

JSON 的类型表示形式如下。

- 数值型（Number）：包括整数和实数。
- 字符串（String）：是由双引号括起来的任意数量 Unicode 字符的集合，它使用反斜杠表示转义字符。
- null：代表“空引用”。
- 布尔型（Boolean）：只有 True 和 False 两个值。
- 数组（Array）：是值（value）的有序集合，它以“[”开始，以“]”结束，值之间使用“,”分隔。
- 对象（Object）：是一个无序的“键/值”映射的集合，它以“{”开始，以“}”结束。每个“名称”后跟一个“:”，集合元素之间使用“,”分隔。其中的值可以

是双引号括起来的字符串（String）、数值（number）、True、False、null、数组或对象，结构可以嵌套。

以下为 JSON 对象的格式：

```
<script type="text/javascript">
    var user = {
            "name":"John",
            "age":23,
            "email":"john@163.com"
    };
  alert(user.name); //也可写成 user["name"];
</script>
```

此时需注意 JSON 格式与对象字面量的区别，JSON 的名字部分严格用双引号将名字括起来，而对象字面量的名字部分则不加引号。

以下增加 Address 属性，并定义成数组，用于表示用户有多个地址。

```
<script type="text/javascript">
    var user = {
      "name":"Mary",
      "age":23,
      "address":
       [
       {"city":"Beijing","zipCode":"111111"},
       {"city":"NanChang","zipCode":"222222"}
       ],
      "email":"Mary@163.com"
    };
    alert(user.name);
    alert(user.address[0].city);   //也可写成 user.address[0]["city"]
    alert(user.address[1].zipCode);
    alert(user.email);
</script>
```

10.4.2　JSON 数据格式解析

JSON 是基于纯文本的数据格式，并且使用 Unicode 编码，这点与 XML 一致。不同的是，XML 比较适合于标记文档，而 JSON 却更适合于实现数据交换处理。

1．在客户端实现 JSON 数据转换

相比 XML 需要从 DOM 中读取各种结点而言，JSON 的使用非常容易。由于 JSON 是 JavaScript 的子集，可直接调用 JavaScript 的 eval 函数将服务器返回的 JSON 文本转化为对象形式。

转化时需要在 JSON 字符串的外面加上一个圆括号：

```
var jsonObject = eval("(" + jsonFormat + ")");
```

加上圆括号的目的是迫使 eval 函数在处理时强制将括号内的表达式转化为对象，而不是作为语句来执行。

现代浏览器，如 Firefox 3.5 和 IE8，它们对 JSON 提供有特殊的支持处理方法，比使用 eval 函数更有效、更安全。

JSON 官方网站上提供有开源的 JSON 解析器和字符串转换器（json.js），包含有如下函数可以生成 JSON 文本。

- string.parseJSON()：将文本数据解析成 JSON 数据。
- object.toJSONString()：将 JSON 数据转换为文本串。该函数的使用对象也可以是其他类型，包括 string、boolean、date、number、array 等。

如果要将 JSON 数据发送到服务器端，则可用如下形式转换处理：

```
var url = "getInfo.asp?people=" + escape(people.toJSONString());
xmlHttp.open("GET", url, true);
xmlHttp.send(null);
```

有许多 AJAX 框架包含了处理 JSON 数据的能力，如 prototype.js（一个流行的 JavaScript 库、http://prototypejs.org）提供了 evalJSON()方法，能直接将服务器返回的 JSON 文本变成 JavaScript 变量。以下为示例代码：

```
new Ajax.Request("http://url", {
    method: "get",
    onSuccess: function(transport) {
        var json = transport.responseText.evalJSON();
        alert(json.xxx);
    }
  }
);
```

2．在服务器端实现 JSON 数据转换

在服务器端中，有的脚本语言提供有专门的 JSON 数据与字符串的转换函数。例如，在 PHP 中可以使用 json_decode()将一串规范字符串解析为 JSON 对象，使用 json_encode()将 JSON 对象生成一串规范字符串。以下为示例代码：

```
<?php
$s = '{"webname":"学练园地","url":"cai.ecjtu.jx.cn"}';
$web = json_decode($s);
echo '网站名称：'.$web->webname.'<br />网址：'.$web->url.;
?>
```

【说明】以上代码中，首先将一个 JSON 规范的字符串赋给变量 s，然后用 json_decode()解析成 JSON 对象，并按照 JSON 对象方式访问对象的数据。

本 章 小 结

本章结合具体应用介绍了 AJAX 技术的编程处理，AJAX 的核心是 XMLHttpRequest 对象的使用，通过 AJAX 可实现无刷新地更新页面内容，从而达到很好的视觉效果。本章介绍了 AJAX 的工作原理、XMLHttpRequest 对象的属性和方法，结合实例讲解了几个应用编程的处理，另外还简要介绍了用于数据传送的 JSON 对象。AJAX 主要用于客户端的脚本编程处理，也可以在服务器端脚本中使用 XMLHttpRequest 对象访问 Internet 的信息服务，从而实现跨服务器的应用集成处理。

习　　题

1．选择题

（1）下列对 JavaScript 的描述，错误的是（　　）。

A．JScript 是 JavaScript 的简称

B．JavaScript 是网景公司开发的脚本语言，其目的是为了简化 Java 的开发难度

C．Firefox 和 IE 存在大量兼容性问题的主要原因在于对 JavaScript 的支持不同

D．AJAX 技术一定要使用 JavaScript 技术

（2）下列关于 AJAX 的描述，正确的是（　　）。

A．AJAX 是一种新兴的技术，是专门用来制作银行网页的

B．AJAX 是一系列技术的集合，主要用到的技术是 Java 技术

C．AJAX 是一种未确定的技术，主要用来进行科学计算

D．AJAX 是异步式 JavaScript 和 XML 的英文缩写

（3）在处理应答中，如果要处理 XML 文档，则需要在参数表中放置 XMLHttpRequest 对象的（　　）属性。

A．responseText　　B．responseXML

C．requestText　　D．requestXML

（4）对于 XMLHttpRequest 对象的 readyState 状态，当 xhr.readyState==1 时表示（　　）。

A．全部取完　　B．正在 load　　C．已经完成

D．未初始化　　E．正在交互

（5）在 AJAX 的 4 种技术中，控制通信的是（　　）。

A．DOM　　B．CSS

C．JavaScript　　D．XMLHttpRequest

2．设计题

（1）本书第 4 章介绍的在线讨论区存在一些缺点，例如，当用户在讨论区中不用超链

接退出，而是关闭浏览器或在浏览器的地址栏输入新的地址访问另外的页面时，系统没有登记到该用户的离开。另外，刷新页面过于频繁，既影响了显示效果，也加重了网络的传输负担，读者可利用目前学到的技术解决这些缺点。

【提示】利用页面的 unload 事件发现用户离开问题；用 setTimeout 函数的定时代替网页标记定义的 refresh 定时刷新。利用 AJAX 技术结合服务器端技术实现讨论区的刷新处理，如果讨论区没有变化信息，则不用更新讨论区页面。

（2）利用 AJAX 实现一个英文单词的查找应用，当用户输入英文单词时，即在页面上显示对应的中文解释。利用数据库表格存储单词的中文解释。

（3）设计一个关键词搜索服务网页，利用服务器端的 XMLHttpRequest 对象尝试访问某个搜索引擎的服务接口，并将结果用自己的方式进行适当处理后显示，也可简单地将返回的内容直接用 response 对象输出。

第 11 章　网络教学综合应用实例

本书作者开发的网络教学系统是一个功能庞大的应用系统，其功能覆盖了教学应用的各个环节。教师可以在该平台的支持下灵活地安排教学内容、组织教学活动、检查教学效果等，同时，学生也可在教师安排下进行学习、操练、讨论、提问、测试、协作等。本章将从该系统中提取几个简单且有代表性的模块来进行介绍。

11.1　网上答疑子系统

网上答疑子系统功能比较简单，其目的是让教师能在网上解答学生提交的问题。本例的系统将学生的问题和教师解答存储在数据库中，是 Web 数据库应用的典型代表。

11.1.1　数据库表格设计

考虑到学生问题和教师解答是一一对应的，所以在系统中只需要设计一个表格即可。该表格的字段说明如表 11-1 所示。

表 11-1　problems 表格设计

字　段	类　型	意　义
id	自动类型	用于唯一性标识学生的问题
title	备注	学生提问的内容
flag_answer	是/否	是否回答
answer	备注	教师解答的内容
time_qry	日期/时间	提交日期/时间
user	文本	提交问题的学生用户标识

11.1.2　辅助包含文件

系统的多个页面采用统一的风格样式，因此可建立一个样式文件存储在 include 子目录下，具体代码如下：

```
-------------------------------------------------- style.inc --------------------------------------------------------------
<style>
BODY{ FONT-FAMILY:  宋体; FONT-SIZE: 14px; LINE-HEIGHT: 20px}
Tr.tr1{ BACKGROUND-COLOR: #FBFDFF }
```

```
Tr.tr2 { BACKGROUND-COLOR: #ffffff }
TD{ FONT-FAMILY: 宋体; FONT-SIZE: 14px;}
input{ FONT-FAMILY: 宋体; FONT-SIZE: 12px;TEXT-DECORATION: none }
A:link {COLOR:#0000ff; TEXT-DECORATION: none}
A:hover {COLOR: #0000ff;    TEXT-DECORATION: none}
A:visited {COLOR:#0000ff; TEXT-DECORATION: none}
</style>
```

【说明】tr 标记定义两个样式是为了实现表格相邻行的风格交替，从而对行与行之间的区分更醒目，同时也增进了网页的视觉美感。

11.1.3　学生端的设计

学生端的功能有两个：一是能看到网上的提问和教师的解答；二是自己能提问。

1．显示学生提问和教师解答

显示学生提问和教师解答内容的界面如图 11-1 所示。本页面的功能是显示所有学生提交的问题和教师解答，其目的是方便学生查看网上已有的问题，同时本程序提供了分页显示功能。另外，对于学生自己的问题可通过超链接转到另一个 ASP 程序，那个程序的界面与此相同，只是提取的数据有差别，在 select 语句中加入了条件过滤。代码如下：

```
strSQL = "select * from problems   where user='"
    &request.cookies("username")&"' order by id desc"
```

其中，Cookie 变量 username 中存储的是用户标识名，在用户登录成功后，系统自动将用户的登录名作为其用户标识名记录在 Cookie 变量中。



图 11-1　分页显示学生提问和教师解答内容的界面

程序 1：分页显示所有提问并提供输入提问表单

```
------------------------------------------------ allproblem.asp ----------------------------------------------------------------
<html><head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<!-- #Include file="include\style.inc" -->
</head><body>
<% strPage = Request.queryString("page")
Set objCnn=Server.CreateObject("ADODB.connection")
objCnn.Open "datasource"       '连接数据源
strSQL = "select * from problems   order by id desc"
set rs=server.createobject("adodb.recordset")
rs.open strSQL,objCnn,1,1
if rs.RecordCount>0 then         '有学生问题时才显示以下内容
'以下根据页跳转请求，设置当前页
rs.PageSize = 6
intPageCount = rs.PageCount
if Len(strPage) = 0 then strPage = "1"
if int(strPage)<1 then strPage = "1"
if int(strPage)>intPageCount then strPage = CStr(intPageCount)
rs.absolutepage=cint(strpage)
%>
<!-- 以下显示分页处理的超链接和表单 -->
<table border="0" cellspacing="1" width="100%">
<tr><td colspan=3 align=center>
<form name="frmSearch" action="allproblem.asp" method="get" >
<a href="problem.asp"><font color=red>▓>></font>自己的问题</a>   
[本页是第<%=int(strPage)%>/<%=intPageCount%>页]
<%  if int(strPage)>1 then
if int(strPage)<>1 then%> [<a href="allproblem.asp?page=1">第一页</a>]
 <% end if %>
[<a href="allproblem.asp?page=<%=CStr( int(strPage)-1 )%>">前一页</a>]
<% end if
if int(strPage)<intPageCount then %>
[<a href="allproblem.asp?page=<%=CStr( int(strPage)+1 )%>">下一页</a>]
<%end if
if int(strPage)<intPageCount then %>
[<a href="allproblem.asp?page=<%=intPageCount%>">最后页</a>]
<%end if %>
  转向第<input type=text class=input id="page" name="page" style='width:20px'
maxlength=3 value="<%=strPage%>">页
<input type='image' src="images/go.gif" ></form>
</td></tr></table>
<table border="0" cellspacing="1" width="100%">
<tr class="tr3"><td width="70" align=center>序号</td>
<td width="*" align=center>提问标题</td>
<td width="150" align="center">提问时间</td> </tr>
<!-- 以下显示本页的问题 -->
```

```
<% dim flagcolor,classtype
flagcolor = true
for num = 1 To rs.PageSize
        if rs.EOF then
                exit for
        end if
flagcolor = not (flagcolor)      '交替切换表格当前行采用的显示样式
if (flagcolor) then
        classtype = "tr1"
else
        classtype = "tr2"
end if %>
<tr class="<%=classtype%>">
<td align=center><b><font color=red><%=num%></font></b></td>
<td><pre><%= rs("title")%></pre>
<%   if (rs("flag_answer")) then
      Response.Write "<pre><font color=#993399><b>答：</b></font>"&rs("answer")&"</pre>"
else
      Response.Write "<font color=#993399><b>未回答</b></font>"
end if    %> </td>
<td align="center"><%=Formatdatetime(rs("time_qry"),1)%> </td> </tr>
<%   rs.MoveNext
Next %>
</table>
<%end if%>
<!--以下为输入问题的表单 -->
<p align=center>
<form method="POST" action="insertproblem.asp">
<font color=green><b>您的问题是：</b></font>
<br><textarea rows="8" name="title" cols="80" wrap=hard></textarea>
<br><br>
<input type="submit" value="提交问题" name="B1">
<input type="reset" value="重新填写" name="B2">
</form></p>
</body>
</html>
```

【说明】编程时要注意边界条件的处理，例如，当没有一个学生问题时，不显示任何内容，但输入问题的表单始终要提供。此时读者应注意琢磨程序中是如何实现表格行的背景色交替更换的。图中翻页的超链接使用了图片，比使用文字更直观。

2．提交新问题

学生可以通过以上查看问题页面最底部提供的“问题输入表单”填写并提交自己的问题，操作界面如图 11-2 所示。



图 11-2　学生输入提问的界面

以下程序将学生输入的问题写入到数据库表格中。

程序 2：将提问写入数据库

```
-------------------------------------------- insertproblem.asp --------------------------------------------------------
<%
strTitle = Trim(Request.Form("title"))
if strTitle<>"" then
strTitle = replace(strTitle,"'","‘")
strTitle = server.HTMLEncode(strTitle)
set conn = server.CreateObject("adodb.connection")
conn.Open "datasource"
strSQL = "insert into problems(title,user) values ('" &strTitle & "','"&request.cookies("username")&"')"
conn.Execute(strSQL)
end if
Server.transfer "allproblem.asp"
%>
```

【说明】

（1）在将问题内容写入数据库表前用 replace 和 server.HTMLEncode 进行了处理，从而保证学生输入的任意内容均能正确显示。前者将单引号换成中文单引号，是避免因为单引号的存在而影响 SQL 语句的格式，后者是保证输入内容中即使含有 HTML 标记也不会影响正常显示。

（2）加入 if 条件判断是为了防止学生将没有内容的问题提交到数据库中。

（3）在写入完毕后用 Server.transfer 转向自己的问题显示页面（problem.asp），从而让该学生能立即看到自己交送的问题。

11.1.4　教师端的设计

教师端的功能应包括浏览问答学生的问题以及对问题的删除管理等操作。

1. 浏览问题

教师浏览学生提问的界面与学生端的界面基本相似，如图 11-3 所示。



图 11-3　教师浏览学生提问的界面

教师浏览学生提问的界面同样提供分页查看功能，所不同的是要保证未回答的问题显示在首页，这点是通过为 select 语句设置查询条件实现的。显示栏目加入了学生姓名，由于 problems 表中只含有登录名，要获取学生姓名则必须通过多表查询，要从 user 表获得学生姓名，所以 select 语句中增加了表间关联表达的条件。另外，由于教师在查看问题的过程中要回答学生问题，并对某些问题进行删除处理。因此，在显示问题时提供了超链接，教师可通过超链接进入问题回答界面。为了实现删除问题处理，在每行前面增加了一个复选框，选中某几行，单击“删除”按钮即可将这些问题删除，为了避免误删，程序中还利用 JavaScript 脚本进一步对删除按钮的 click 事件进行确认。

程序 3：分页浏览提问

```
---------------------------------------- problemadmin.asp -----------------------------------------------------------
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<!-- #Include file="include\style.inc" -->
</head>
<body background="images\bg.gif">
<% strPage = Request.queryString("page")
content = Request.queryString("content")
Set objCnn=Server.CreateObject("ADODB.connection")
objCnn.Open    "datasource"
'以下为了处理基于内容搜索学生提问
```

```
if content<>"" then
  strSQL = "select id,title,flag_answer,user,username from problems,user where problems.user=user.
loginname and problems.title like '%"&content&"%' order by  not flag_answer, id desc "
else
  strSQL = "select id,title,flag_answer,user,username from problems, user where problems.user=
user.loginname order by  not flag_answer, id desc "
end if
set rs=server.createobject("adodb.recordset")
rs.open strSQL,objCnn,1,1%>
<table border="0" cellspacing="1" width="100%">
<tr><td colspan="4">
<%if rs.eof then %>
     <p>目前无任何问题。</p></td></tr></table>
<%  response.end
end if
sumcount = rs.RecordCount
rs.PageSize = 12
intPageCount = rs.PageCount
if Len(strPage) = 0 then strPage = "1"
if int(strPage)<1 then strPage = "1"
if int(strPage)>intPageCount then strPage = CStr(intPageCount)
rs.absolutepage=cint(strpage)
%>
<form name="frmSearch" action="problemadmin.asp" method="get" >
<p align="center">[本页是第<%=int(strPage)%>/<%=intPageCount%>页]
<% if int(strPage)>1 then
%> [<a href="problemadmin.asp?page=1">第一页</a>]
[<a href="problemadmin.asp?page=<%=CStr( int(strPage)-1 )%>">前一页</a>]
<%end if
if int(strPage)<intPageCount then
%> [<a href="problemadmin.asp?page=<%=CStr( int(strPage)+1 )%>">下一页</a>] [<a href=
"problemadmin.asp?page=<%=intPageCount%>">最后页</a>]<%
          end if%>  转向第<input type=text  name="page"  style='width:20px'
maxlength=3 value="<%=strPage%>">页  搜索问题为：<input type=text name="content"
style='width:80px' maxlength=8 ><input type='image' src="images/go.gif" ></p></form>
<form action="problemdelete.asp" method="POST" >
<table width=100%>
<tr class="tr3"><td width="5%" ALIGN="CENTER">选中</td>
<td width="55%" align="center">标题（点击标题可解答问题）</td>
<td width="8%" align="center">登录名</td>
<td width="10%" align="center">姓  名</td>
<td width="8%" align="center">回答否</td></tr>
<%
dim flagcolor,classtype
flagcolor = true
for intRecord = 1 To rs.PageSize
     if rs.EOF then
          exit for
     end if
```

```
    flagcolor = not (flagcolor)
        if (flagcolor) then
            classtype = "tr1"
        else
            classtype = "tr2"
        end if
%>
<tr class="<%=classtype%>">
<td width="5%" ALIGN="CENTER">
<input TYPE="CHECKBOX" NAME="id" VALUE="<%=rs("id")%>"></td>
<td width="55%">
<pre><a href="problem_detail.asp?ID=<%=rs("ID")%>&page=<%=strpage%>"><%= rs("title")%></a>
</pre></td>
<td width="8%" align=center><%response.write rs("user")%> </td>
<td width="10%" align=center><%response.write rs("username")%> </td>
<td width="8%" align=center>
<% if rs("flag_answer") = true then
        Response.Write "yes"
else
        Response.Write "no"
end if   %>
</td></tr>
<% rs.MoveNext
Next %>
<tr><td colspan="4">
<input type="hidden" name="page" value="<%=strpage%>">
<p align=center>
<input type="submit" value=" 删 除 " name="btndel" onclick="return confirm('确定删除所选问题吗？');">
</p></td></tr>
</table>
</form>
</body>
```

【注意】

（1）本例中还包括一个功能，即基于学生问题内容进行搜索查找，这里利用 SQL 的 like 运算符实现模糊查找处理，只要搜索问题的内容中含有要查找的内容即可显示出来。

（2）需注意边界条件的处理，当数据库中没有一个学生问题时，显示出一个信息，然后利用 response.end 方法直接停止执行程序，这样逻辑上就显得十分清晰。

（3）注意体会显示问题时为后续操作提供导航，首先是回答问题，针对每个问题设计相应的超链接转到回答问题的输入界面，但注意要为超链接提供 URL 参数。参数的提供要根据后续页面处理的需要来设置。另一操作是删除问题，在浏览页面提供有表单，针对每个问题提供一个复选框，选中的问题就是将要被删除的对象，特别要注意的是，复选框的名称和值如何与问题对应。这里是所有复选框用同一个名字，将问题标识作为相应复选框的值，这样在后续页面中根据该值可知道要删除哪些问题。

2．回答问题

在回答问题页面要显示学生问题，注意超链接传递过来的两个参数，一个参数是问题标识（ID），用于查找问题；另一个参数是 page，用于在回答完毕后返回指定的页。教师解答学生提问的界面如图 11-4 所示。

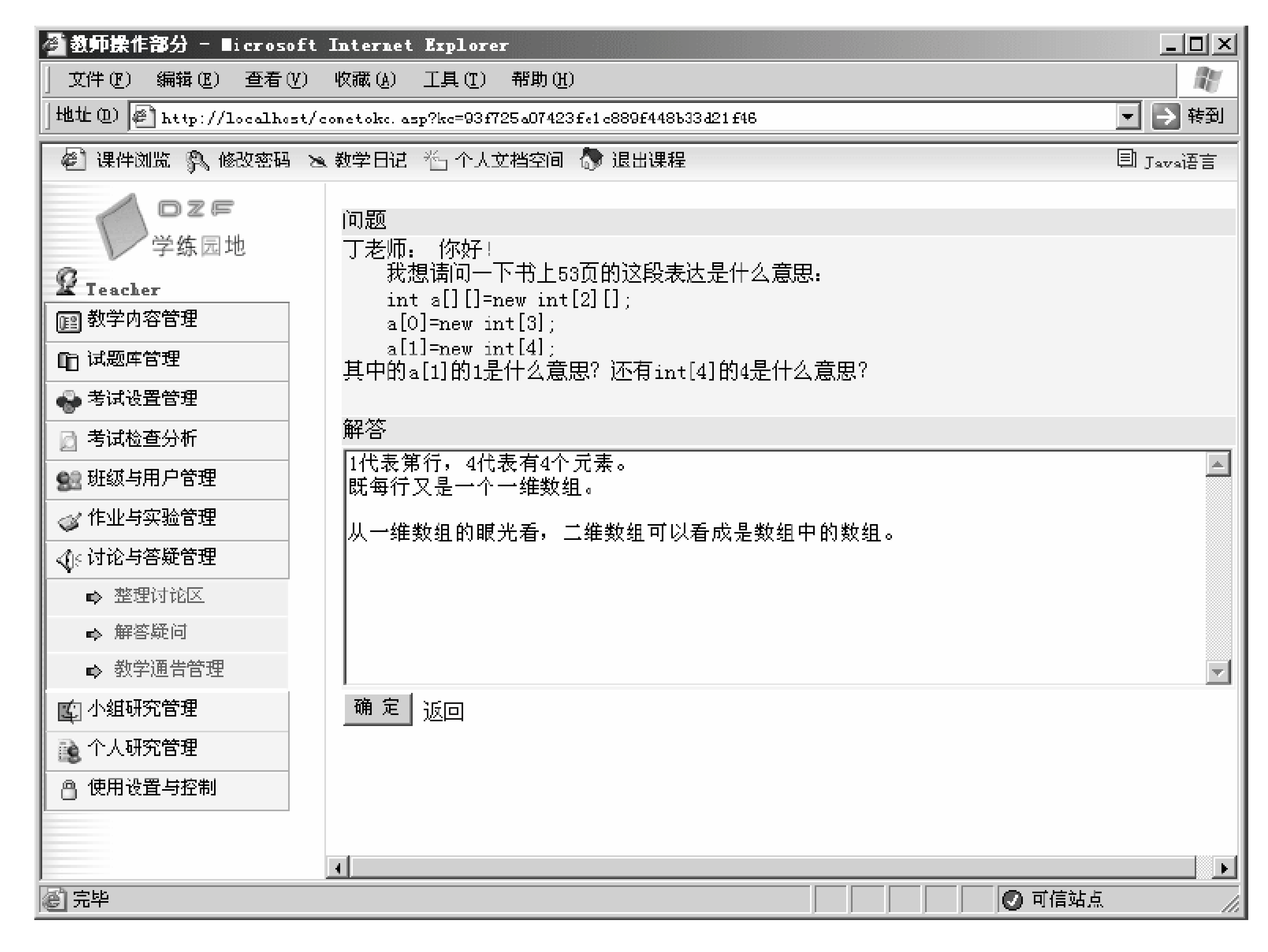


图 11-4　教师解答学生提问的界面

程序 4：输入解答界面

```
----------------------------------------------- problem_detail.asp -----------------------------------------------------------
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<!-- #Include file="include\style.inc" -->
</head>
<body>
<% strID = Request.queryString("ID")
strpage =        Request.queryString("page")
Set objCnn = Server.CreateObject("ADODB.Connection")
objCnn.Open    "datasource"
  '根据问题标识查找问题
strSQL = "SELECT id,title,answer FROM problems where ID="&strID
Set rs = Server.CreateObject("ADODB.Recordset")
rs.open strSQL,objCnn,1,1
%>
<form action="problem_answer.asp" method="POST" >
<table border="0" cellspacing="1" width="100%">
<input type="hidden" id="id" name="id" value="<%= strID %>">
<input type="hidden" id="page" name="page" value="<%=strpage%>">
<tr class="tr3"><td>问题</td></tr>
<tr class="tr1"><td><pre><%= rs("title")%></pre></td></tr>
```

```
<tr class="tr3"><td>解答</td></tr>
<tr class="tr1"><td>
<textarea rows="10" name="answer" cols="80"><%=rs("answer")%></textarea>
</td></tr>
<tr><td><p><input type="submit" value="确  定" id="ok" name="ok"> 
<a href="problemadmin.asp?page=<%=strpage%>">返回</a></p></td></tr>
</table>
</form>
</body>
</html>
```

【说明】在输入表单中应注意两点：第一点是如果该问题教师已解答，则要将已有解答显示出来，另外教师还可以更改解答；第二点是通过隐含域将问题 ID 和 page 参数传递给后续处理程序，请读者思考：如果将隐含域改为用 URL 参数传递的办法，应如何表示，在后续页面中又该如何读取。

程序 5：将解答写入数据库

```
---------------------------------------------- problem_answer.asp ------------------------------------------------
<%
id = Request.Form("id")
strAnswer = Request.Form("answer")
strAnswer = replace(strAnswer,"'","‘")
set conn = server.CreateObject("adodb.connection")
conn.Open "datasource"
strSQL = "update problems set flag_answer=true,answer = '" & strAnswer & "' where id="&id
conn.Execute(strSQL)
response.redirect "problemadmin.asp?page="& Request.Form("page")
%>
```

【说明】回答处理问题非常简单，只需使用 update 语句更新数据库表，将对应问题的已回答标记字段（flag_answer）置为 True，并将解答字段（answer）更新即可。

【思考】通过隐含域传递 page 的作用，重定向的 URL 不跟参数有何不足。

3．删除学生问题

```
---------------------------------------------- problemdelete.asp ---------------------------------------------------
<% strpage = Request.Form("page")
sqlpre = "DELETE FROM problems WHERE id = "
set conn = server.CreateObject("adodb.connection")
conn.Open "datasource"
for i = 1 to request.form("id").count
	conn.Execute sqlpre & request.form("id")(i)
next
response.redirect "problemadmin.asp?page="&strpage
%>
```

【说明】删除学生处理程序是一个典型的批量处理的范例，在查询页面中每个问题对应有一个名字均为 id 的复选框，这些复选框的值分别为对应的问题标识。在此程序中

request.form("id")所得到的是一个集合，集合中的每个元素即为选中的问题对应的问题标识，也就是要将这些问题删除，此时即可通过一个循环实现，delete 语句的前面部分相同，只是后面的条件部分有变化。

【思考】for 循环改用 for each 实现应如何修改程序？

11.2　基于 XML 的单元自测应用

11.2.1　功能概述

基于 XML 的单元自测应用的目标是在网络课件中为每个单元安排一份自测试卷，学生可以解答该份试卷，单击“交卷”按钮，程序将自动根据试卷中的标准答案对学生进行评分，将得分告诉学生，并显示各题的标准答案和学生解答，以便学生对比检查。该应用完全是一个客户方浏览器上的交互应用，综合体现了客户方编程技术的运用。试卷存储在 XML 文件中，由 JavaScript 根据当前学习的单元装载相应的试卷，然后采用 XML DOM 技术和 XSL 变换技术访问 XML 试卷内容，并利用 DHTML 技术生成相应的试卷解答界面。依据事件代码实现解答登记和评分处理，并将解答对比显示给学生。单元测试做题界面如图 11-5 所示。

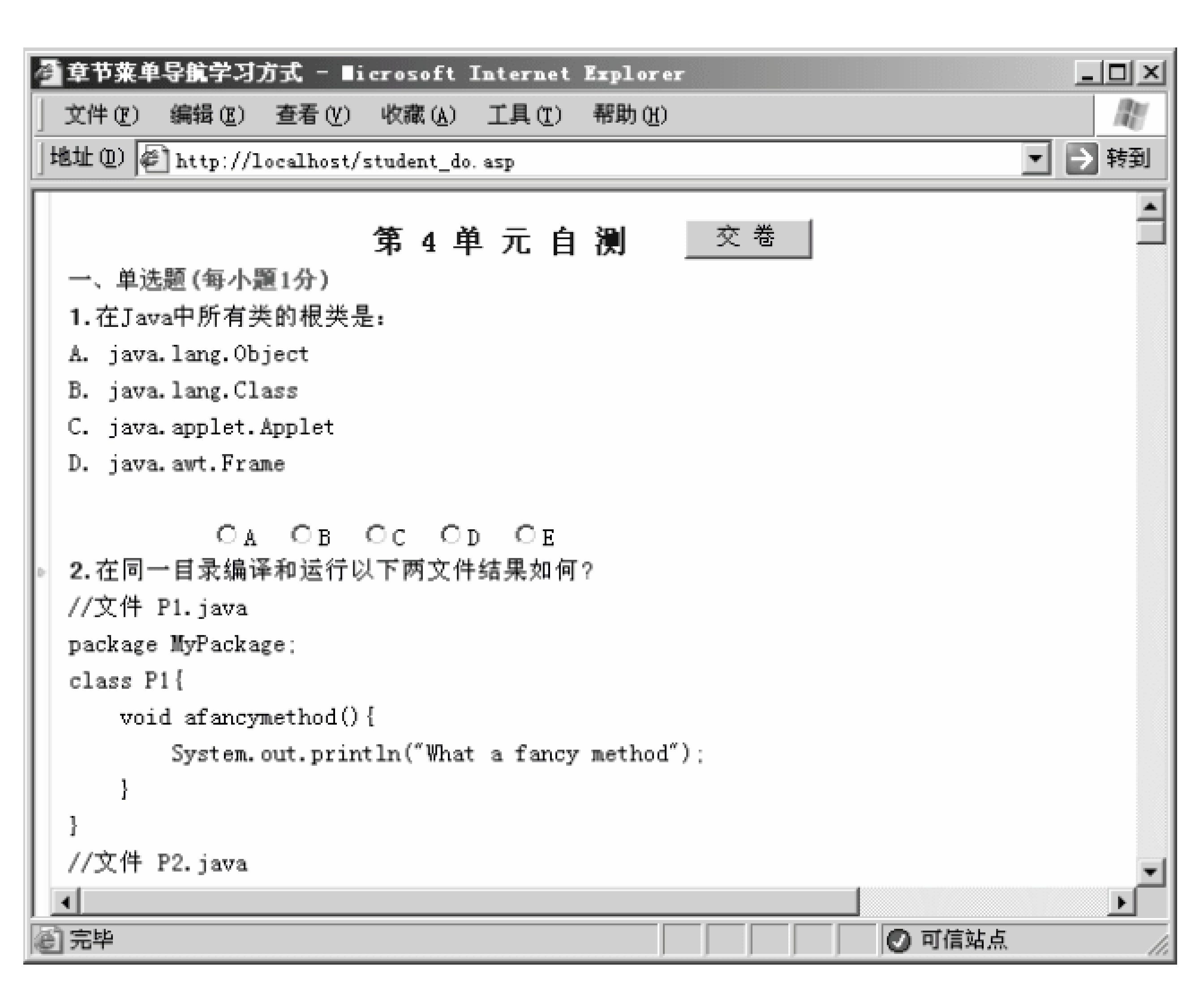


图 11-5　单元测试做题界面

11.2.2　测试试卷的 XML 表示

为了节省篇幅，这里只给出了 3 类试题，且每类试题只给出一道题作为样例。整个试卷的根结点为 paper，各类试题的相关标记的含义如表 11-2 所示。

表 11-2　试卷中 XML 标记的含义

试 题 类 型	标　　记	含　　义
单选题	singlechoice	单选题类的根结点
	score	每道试题的分数
	shiti	一道试题的根结点
	content	本道试题内容
	answer	本道试题标准答案
多选题	multichoice	多选题类的根结点
	score	每道试题的分数
	timu	一道试题的根结点
	content	本道试题内容
	answer	本道试题标准答案
多项填空题	integrity	多项填空题类的根结点
	score	每空分数
	material	阅读材料
	mycontent	材料中一段内容
	wk	试题内容中一个空

以下为 XML 试卷文件样例（exam3.xml）

```
<?xml version="1.0" encoding="gb2312"?>
<paper examtime="100">
<singlechoice>
<score>1</score>
<shiti>
<content>
如何定义一个不能有子类的类 Key？
  A.   class Key { }
  B.   abstract final class Key { }
  C.   native class Key { }
  D.   final class Key { }
</content>
<answer>D</answer>
</shiti>
<shiti>……</shiti>   <!--下一道试题 -->
</singlechoice>
<multichoice>
<score>1</score>
<timu>
<content>
下列哪些是 Java 中合法的修饰符？
A. private
B. public
C. protected
D. protect
```

```
</content>
<answer>ABC</answer>
</timu>
<timu>.......</timu>   <!--下一道试题 -->
</multichoice>
<integrity>
<score>2</score>
<material>
<mycontent>public class Java_3
{      int x,y; //点的坐标
       public Java_3() { }
       public Java_3(int x,int y){<wk>this.x=x;this.y=y;</wk>}
       public Java_3(Java_3 p){<wk>x=p.x;y=p.y;</wk>}
}</mycontent>
<mycontent>……</mycontent>    <!--下一段落 -->
</material>
<material>……</material>   <!--下一道试题 -->
</integrity>
</paper>
```

【说明】由于一类试题的各道题的分数相同，所以 Score 标记作为试题类的子结点，而没有安排作为每道试题的子结点。多项填空题考虑到一定的通用性要求，每道试题为一份阅读材料（material），其中包含若干段落（mycontent），在每段中安排一些空（wk），各空的标准答案直接在 wk 标记中。

11.2.3　考试解答界面的生成及显示处理

1．试卷的装载

首先定义一个函数 getData，该函数将在页面加载时由 OnLoad 事件触发执行，它将完成试卷 XML 文件的加载，并通过对 XML 文件的分析处理生成试卷解答界面的 HTML 代码，最后用 DHTML 技术将解答界面的 HTML 代码显示在页面中。

```
function getData( ){
   xmldoc = new ActiveXObject("Microsoft.XMLDOM");
   xmldoc.async = false;
   url1="exam"+zh+".xml";                    //zh 为当前学习的章号
   xmldoc.load(url1);
   root = xmldoc.documentElement;            //根结点
   ……              //以下生成试卷解答的 HTML 代码字符串
```

2．各类试题解答界面的生成处理

在各类试题的解答界面生成处理中，一方面要显示试题内容，另一方面要生成相应的解答控件，同时要解决解答控件的命名和相应的事件处理问题。所有内容的显示处理均通过字符串的拼接来完成，要显示的内容保存在 res 字符串变量中。

（1）单选题解答界面的生成处理

```
res="一、单选题</font><font color='green'><b>(每小题"
res+= root.selectSingleNode("singlechoice/score").text+"分)</b></font><td>"
items= root.getElementsByTagName("singlechoice/shiti/content")
ct=0     //记录试题序号
for (i=0;i<items.length;i++) {
  ct=ct+1;
  res+= "<tr><td colspan='3'><pre><font face='黑体' color='blue'><b>"
  res+= ct+"</b></font>."+items(i).text+"</pre></td></tr><tr>"
  res+= "<td width='10%'></td><td colspan='2' width='90%'>"
  res+= "<input type='radio' value='A' name='x"
  res+= ct+"' onclick='log(this,"+ct+")'>A  "
  res+= "<input type='radio' value='B' name='x"+ct
  res+= "' onclick='log(this,"+ct+")'>B  "
  res+= "<input type='radio' value='C' name='x"+ct
  res+= "' onclick='log(this,"+ct+")'>C  "
  res+= "<input type='radio' value='D' name='x"+ct
  res+= "' onclick='log(this,"+ct+")'>D  "
  res+= "<input type='radio' value='E' name='x"+ct
  res+="' onclick='log(this,"+ct+")'>E"
  res=res+"</td></tr>"
}
```

【说明】在单选题的解答界面生成处理中有两个要点：① 控件的命名，这里采用 x 作为前缀后跟试题序号的办法，注意同一题的各单选按钮名称相同，如此才能达到单选的目的。② 事件的处理，如何将用户选择的解答通过事件操作记录下来，以便评分处理。这里采用了传递试题序号和代表按钮控件对象本身的变量 this 的办法实现解答选择信息的传递，试题序号知道做的是哪道题，而 this 参数则可以知道点击了哪个选项，从而可以根据它获取选项的值。

（2）多选题解答界面的生成处理

```
res+="<p><font color='blue'><b>二、多选题</font><font color='green'>"
res+=" (每小题"+root.selectSingleNode("multichoice/score").text
res+="分)</b></font></p>"
items=root.getElementsByTagName("multichoice/timu/content")
ct=0
for (i=0;i<items.length;i++) {
  ct=ct+1;
  res+= "<p><pre><font face='黑体' color='blue'><b>"+ct
  res+= "</b></font>."+items(i).text+"</pre></p>"
  res+= "<input type='checkbox' value='A' name='m1_"+ct
  res+= "' onclick='logmultichoice(this,1,"+ct+")'>A  "
  res+= "<input type='checkbox' value='B' name='m2_"+ct
  res+= "' onclick='logmultichoice (this,2,"+ct+")'>B  "
  res+= "<input type='checkbox' value='C' name='m3_"+ct
  res+= "' onclick='logmultichoice (this,3,"+ct+")'>C  "
  res+= "<input type='checkbox' value='D' name='m4_"+ct
```

```
    res+= "' onclick='logmultichoice (this,4,"+ct+")'>D  "
    res+= "<input type='checkbox' value='E' name='m5_"+ct
    res+= "' onclick='logmultichoice (this,5,"+ct+")'>E"
}
```

【说明】多选题的控件命名与单选题类似，只是同一题的各个复选框不同名，主要是解答处理函数传递的信息要 3 个参数，第一个参数是 this，代表选择的那个复选框对象；第二个参数是选项的序号，以便登记时知道选择了哪个选项；第三个参数是试题序号。

（3）多项填空题解答界面的生成处理

多项填空题将针对试题中的空的位置生成相应的文本框供用户填写解答，处理时涉及空的查找与转换，利用 XSL 变换处理可以较容易地进行处理。

```
res+= "<p><font color='blue'><b>三、多项填空题</font><font color='green'>"
res+= " (每空"root.selectSingleNode("integrity/score").text
res+= "分)</b></font></p>"
root = root.selectSingleNode("integrity")          //选取多项填空类试题的根结点
xsldoc = new ActiveXObject("Microsoft.XMLDOM");
xsldoc.async = false;
xsldoc.load("exam2.xsl");                          //装载 XSL 变换程序
x = root.transformNode(xsldoc);                    //对整个多项填空类试题进行变换并返回变换结果
res = res+x;                                       //将处理结果拼接到 HTML 显示串 res 中
```

以下为相应的 XSL 变换程序（exam2.xsl）

```
---------------------------------------------------- exam2.xsl ------------------------------------------------------------
<?xml version  ="1.0"  encoding="GB2312"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl" languange="JavaScript">

<!-- 模板 1 -->
<xsl:template match="integrity">
<xsl:eval>var st=0;""</xsl:eval>
<xsl:eval>var ct=0;""</xsl:eval>
<xsl:for-each select="material">
<tr><td colspan='3'>
<xsl:apply-templates select="mycontent"/>
</td></tr>
</xsl:for-each>
</xsl:template>

<!-- 模板 2 -->
<xsl:template match="mycontent">
<pre><b><font color='#800000'>
<xsl:eval>st=st+1;"试题"+st+":"</xsl:eval></font></b>
<xsl:apply-templates select="wk|br|text()"/></pre>
</xsl:template>

<!-- 模板 3 -->
<xsl:template match="wk">
<u><b><xsl:eval>ct=ct+1;" ("+ct+") "</xsl:eval></b></u>
```

```
<input type="text" size="16">
<xsl:attribute name="name">w<xsl:eval>ct</xsl:eval>
</xsl:attribute>
<xsl:attribute name="onchange">logkong(this,<xsl:eval>ct</xsl:eval>)
</xsl:attribute>
</input>
</xsl:template>

<!--  模板 4 -->
<xsl:template match="br"><br/></xsl:template>

<!--  模板 5 -->
<xsl:template match="text()"><xsl:value-of/></xsl:template>
</xsl:stylesheet>
```

【说明】这部分的处理有一定难度，为了实现将多项填空题中的空用特殊的输入框显示，程序中共定义了如下 5 个模板：

① 模板 1 定义了多项填空类试题的根结点的处理策略，其中定义了两个变量，一个是 st 代表试题序号，另一个是 ct 代表空的序号。模板中对每道多项填空题（material）通过 for each 循环进行匹配处理，每份阅读内容（mycontent）调用模板 2 进行处理。

② 模板 2 首先计算输出试题的序号，然后对其中的内容（包括文本和填空项）分别调用模板 3、模板 4 和模板 5 进行处理。

③ 模板 3 用于填空项的匹配处理，它包括计算和显示一个填空项的序号。另一个难点是产生输入文本框，文本框中的属性通过 xsl:attribute 标记进行定义，标记名用 m 作为前缀后跟空的序号。onchange 事件属性定义的函数 logkong 包括两个参数，一个是代表文本框对象的 this 参数，另一个是空的序号。

④ 模板 4 实现对 XML 文本中
标记的特殊处理，以保证
标记按换行处理。

⑤ 模板 5 用于文本的匹配处理，它只是将文本的内容照原样输出。

多项填空题变换后的页面显示效果如图 11-6 所示。

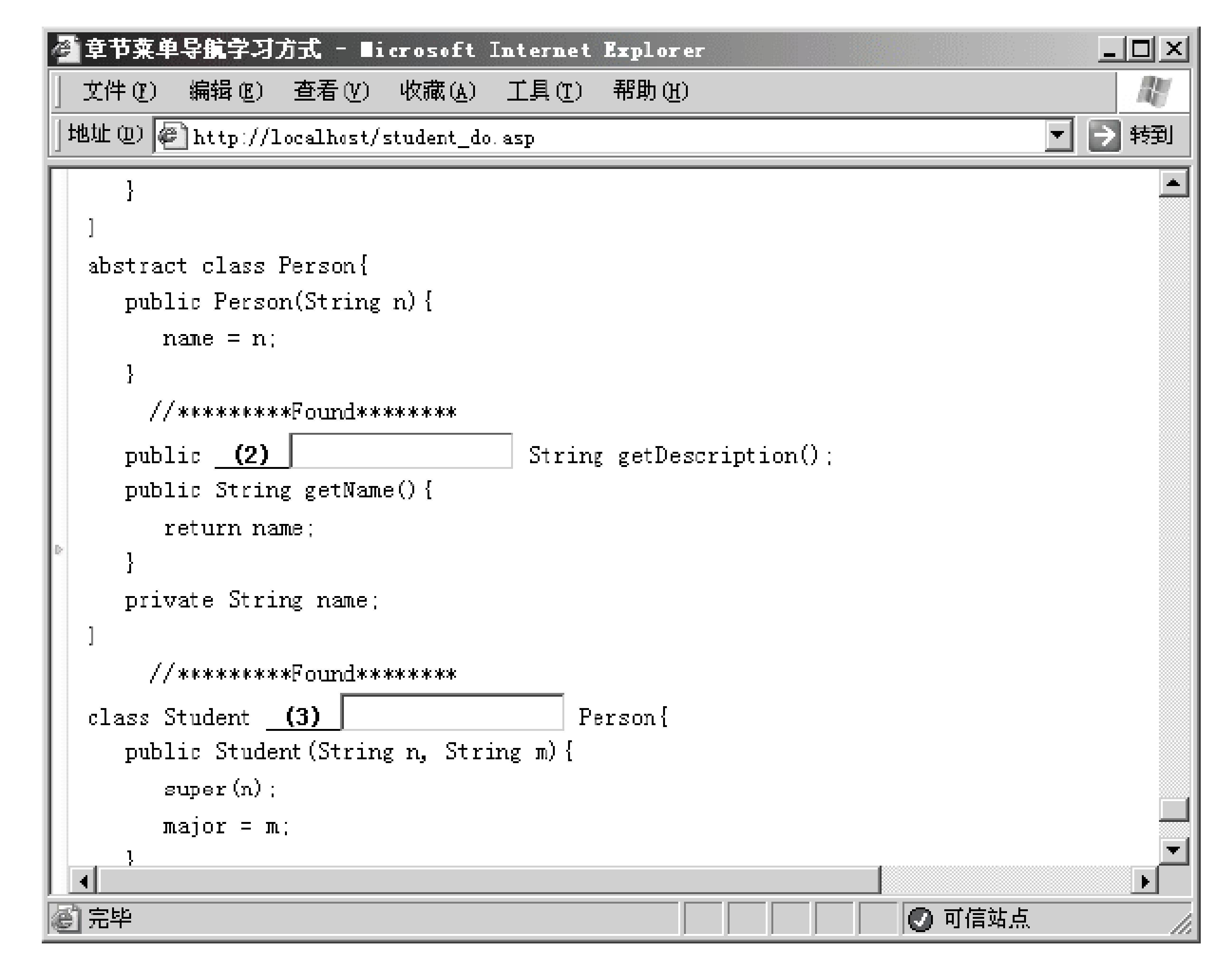


图 11-6　多项填空题变换后的页面显示效果

3．解答界面的显示处理

函数 getData 的最后一段脚本代码以及页面显示层的定义代码如下（脚本中利用 DHTML 技术将生成的显示字符串（res）写入页面中标识名为 me 的层中，从而实现动态显示）：

```
<!--在试卷的开头和结尾均安排了交卷按钮，开头部分省略了代码 -->
res+= "<p>  <input type='button' value=' 交 卷 '"
res+= " onclick='checkresult( )' ></p>"
me.innerHTML = res;   //将拼接好的解答界面的 HTML 串写入名字为 me 的层中
}
</script >
<body   onload="getData( )">
<div id="bt"></div>     <!-- 显示试卷标题的层 -->
<div id="me"></div>     <!-- 显示试题解答界面的层 -->
</BODY>
```

【说明】试卷标题层的显示在这里未介绍，它只是将第几单元测试的文字显示出来，单元的值取决于用户章节导航中选择了哪章，需要通过信息传递得到。

11.2.4　考试的解答记录、交卷评分及答案对比的显示

1．通过数组记录解答

通过数组记录用户对各题的解答选择，单选题和多项填空题的解答均用一维数组记录，多选题用二维数组记录，JavaScript 不能直接定义二维数组，而应通过定义数组的数组实现。

```
var arr=new Array(100);                  //单选题
var multich=new Array(50);               //多选题
for (k=0;k<arr.length;k++)
     multich[k]=new Array(5);            //多选题的 5 个选项
var kong=new Array(30);                  //多项填空题
```

2．记录解答的函数

将用户的解答选择记录到数组中，这样在判分处理时将更快捷。通过用户答题时触发的事件而执行相应的函数处理。单选题和多选题的记录分别是依托选项按钮和复选框的 onclick 事件，而多项填空题的记录则是依托文本框的 onchange 事件。

（1）单选解答记录函数

```
function log(m,i){                       //第 i 道题
    arr[i]=m.value;
}
```

（2）多项填空解答记录函数

```
function logkong(m,i){                   //第 i 个填空题
    kong[i]=m.value;
}
```

（3）多选解答记录函数

```
function logmultichoice(m,j,i){   //第 i 道题的第 j 个选项
      if (m.checked)
         multich[i][j-1]=m.value;
      else
         multich[i][j-1]="";
}
```

3．交卷处理函数

单击“交卷”按钮将触发执行 checkresult 函数，该函数将访问 XML 文件读取每道试题的标准答案与数组中记录的用户解答进行对比，进而计算得分。为了将解答对比和得分告诉用户，和前面的生成试卷解答页面类似，这里同样要进行显示内容的拼接处理，并在拼接后的 HTML 文本中显示试题内容、标准答案和用户解答，如图 11-7 所示。计算的得分通过对话框提示给用户。



图 11-7　答案对比显示界面

以下为函数代码：

```
function checkresult ( ){
score=0;                          //用于累计用户得分
totalscore=0;                     //用于累计试题总分值
str="<center><strong><font color='red'>(答案对比)</font></strong>"
str=str+"</center><br><table><td width=5></td><td>"
/*  以下为单选题的解答对比处理  */
s=xmldoc.selectSingleNode("//singlechoice/score");
xscore=parseInt(s.text);          //单选题每题分值
ans=xmldoc.selectNodes("//singlechoice/shiti");
```

```
no=1;                                          //试题序号
mynodes=ans.length;                            //单选题数量
str=str+"<font color='blue'><b>一.单选题</b></font><br>";
for (k=0;k<mynodes;k++) {
  m=arr[no];
  x=ans.item(k).childNodes.item(1).text;        //获取试题标准答案
  totalscore=totalscore+xscore;
  str +="<pre><b>"+(k+1)+"</b>."
  str += ans.item(k).childNodes.item(0).text+"</pre><br>";
  if (m==x) {
    score=score+xscore;
    str+= "答案："+x+"，你的解答："+m+"<br>";
  } else
    str+= "答案：<font color='red'>"+x+"</font>,你的解答："+m+"<br>";
  no=no+1;
}
/* 以下为多选题的解答对比处理 */
s=xmldoc.selectSingleNode("//multichoice/score");
xscore=parseInt(s.text);
ans=xmldoc.selectNodes("//multichoice/timu");
no=1;
mynodes=ans.length;
str+=+"<br><font color='blue'><b>二.多选题</b></font><br>";
for ( k=0;k<mynodes;k++) {
  m="";
  for(d=0;d<5;d++) m=m+multich[no][d];          //从数组取得用户解答
  x=ans.item(k).childNodes.item(1).text;
  totalscore=totalscore+xscore;                 //累计试题总分值
  str += "<pre><b>"+(k+1)+"</b>."
  str += ans.item(k).childNodes.item(0).text+"</pre><br>";
  if (m==x) {
    score=score+xscore;
    str+= "答案："+x+"，你的解答："+m+"<br>";
  }
  else
    str+= "答案：<font color='red'>"+x+"</font>,你的解答："+m+"<br>";
  no=no+1;
}
/* 以下为多项填空题的解答对比处理 */
s=xmldoc.selectSingleNode("//integrity/score");
xscore=parseInt(s.text);
str += "<br><font color='blue'><b>三.多项填空题</b></font><br>";
xsldoc = new ActiveXObject("Microsoft.XMLDOM");
xsldoc.async = false;
xsldoc.load("exam3.xsl");
root=xmldoc.selectSingleNode("//integrity");
res1=root.transformNode(xsldoc);
str=str+res1;
```

```
ans=xmldoc.selectNodes("//wk");
no=1;
mynodes=ans.length;
for ( k=0;k<mynodes;k++) {
   m=kong[no];
   x=ans.item(k).text;
   totalscore=totalscore+xscore;                        //累计试题总分值
   if (m==x) {
         score=score+xscore;                            //解答正确，累计得分
         str += "<font color='blue'><b>空("+(k+1)+")</b></font>.答案："+x
         str += "，你的解答："+m+"<br>";
   }else {
         str +="<font color='blue'><b>空("+(k+1)+")</b></font>.答案："
         str +="<font color='red'>"+x+"</font>，你的解答："+m+"<br>";
         no=no+1;
   }
   str=str+"</td></table>";
   score=Math.floor(score*100/totalscore);              //折算为百分制分数
   alert("考试分数（折算为百分数）："+ score);          //告诉用户得分
   me.innerHTML=str;                                    //将答案对比显示出来
}
```

【说明】多项填空题显示处理仍用到 XSL 变换，和前面的处理基本一样，只是对空的处理做了变化，不再显示输入框，而是显示一个前后加小括号的空符号，并加有下划线表示。以下为相应模板的代码：

```
<xsl:template match="wk">
<u><b><font color='red'><xsl:eval>ct=ct+1;" ("+ct+") "</xsl:eval></font></b></u></xsl:template>
```

11.3　网络课件导航菜单的设计

导航菜单是网络课件中使用最频繁的部件，它的设计既要考虑用户使用的方便性，又要考虑快速响应的特点，同时还要设法减轻服务器和网络的负担。

11.3.1　导航菜单的设计要求

学生学习的操作页面由 3 帧构成：一是系统功能图标区；二是导航菜单；三是内容显示区，如图 11-8 所示。导航菜单的处理包括两部分：一是要根据选择的章、节、问题逐级展开目录树，二是要根据用户的选择显示网页内容。导航菜单初始打开的章为教师指定的“当前章”，用户点击目录树的结点将根据该结点的位置生成新的目录树，同时在内容显示区显示该结点对应的教学页面。

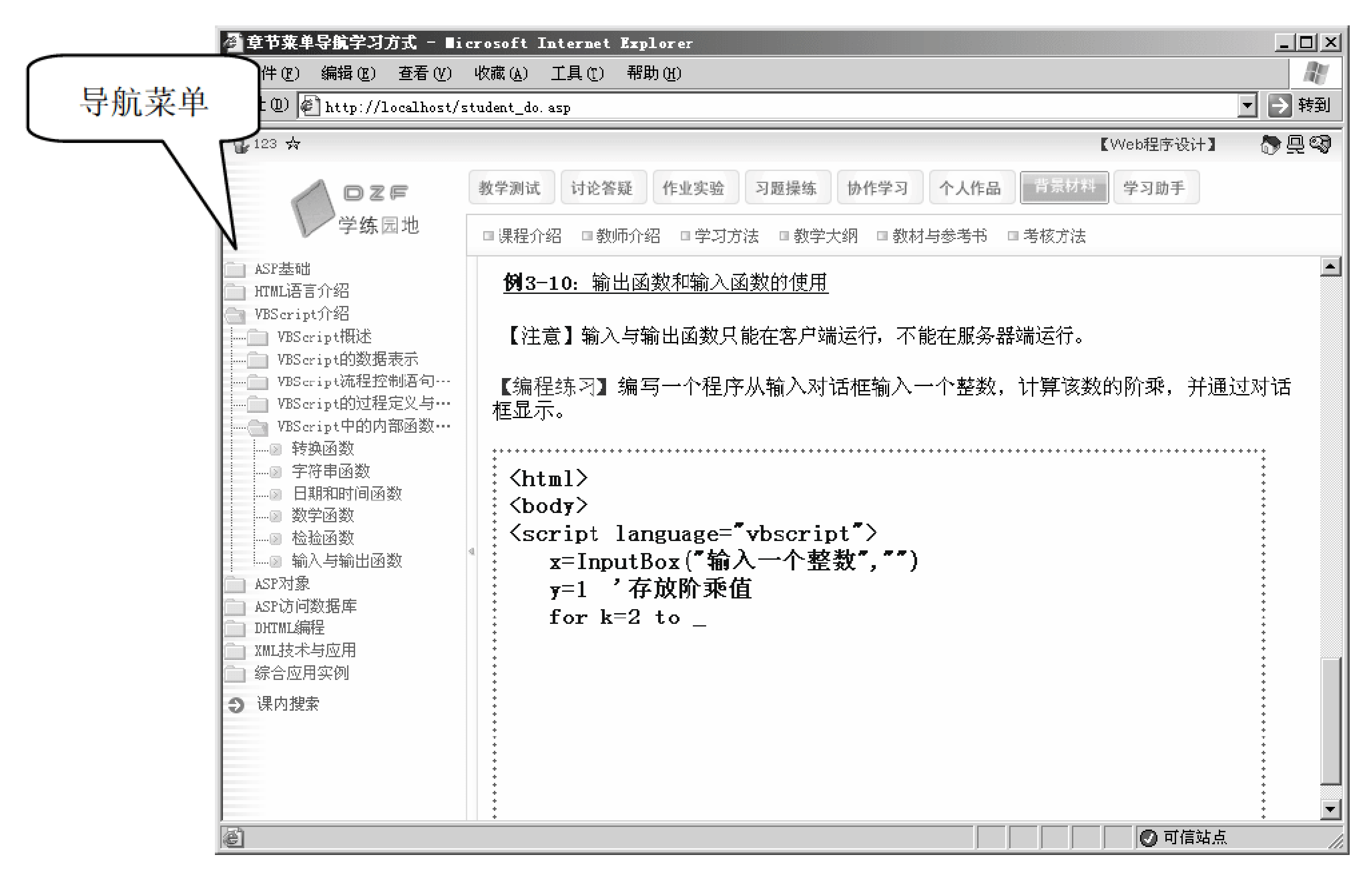


图 11-8　学生主页面帧的划分

11.3.2　基于 XML 的导航菜单设计

基于 XML 的导航菜单的设计思路是将与章节导航相关的信息生成 XML 表示形式，将 XML 数据传递到客户端，由 JavaScript 代码访问 XML 数据，并利用 DHTML 技术实现导航菜单目录树的动态变化。

1．章节导航信息的 XML 文本格式

以下为表示章节导航信息的 XML 文本格式，章结点、节结点、问题结点按层次结构进行组织。每章下面包含若干节，每节包含若干问题。所有表示章、节、问题的结点均用 node 标识，每个 node 结点包括标题（title）、结点内容文件名（url）两个属性。title 用于导航菜单各级标题的显示，url 为结点对应的 HTML 文件名，形式为“text 章号-节号-问题号.htm”。

存放章节层次结构的 XML 文件的格式如下：

```
---------------------------------------------- chapmenu.xml------------------------------------------------------
<?xml version='1.0' encoding='gb2312'?>
<directory>
<node title="JAVA 概述" url="text1-0-0.htm" >          <!--章结点-->
  <node title=">面向对象概述"url="text1-2-0.htm">          <!--节结点-->
    <node title=">面向对象与面向过程区别"url="text1-2-1.htm "/>
      <!--问题结点-->
      ……  <!--其他问题-->
        </node>
         ……  <!--其他节-->
    </node>
```

```
      ……   <!--其他章-->
</directory>
```

2. 菜单显示处理

以下程序利用 JavaScript 脚本装载访问 XML 文档，并通过对文档的分析生成导航菜单。这里的关键是 list 函数的设计，在 list 函数中包括 3 个参数，分别表示章、节、问题，该函数的功能是根据指定的参数显示导航菜单，并在内容显示帧中显示对应当前结点的 HTML 内容。页面初始装载时，利用 onload 事件让菜单默认打开第 1 章内容（text1-0-0.htm）。list 函数的代码包括两部分，第一部分是获取当前结点对应的 URL 文件名，并在内容显示区显示对应的页面内容；第二部分是根据章节参数生成用来显示导航菜单的 HTML 字符串，并将生成的导航菜单显示在 menu 层定义的位置，导航菜单内容按表格形式进行排列，菜单中显示每条标题的单元格定义了 onclick 事件让其调用 list 函数实现与指定章节的关联。

```
<html><head>
<script type="text/javascript">
var root;
var kcpath="<%=request.cookies("path")%>";   //获得课程内容的存储路径

function init() {
    var xmldso = new ActiveXObject("Microsoft.XMLDOM");
    xmldso.async="false";
    xmldso.load("chapmenu.xml");
    root=xmldso.documentElement;                  //XML 文档的根结点
    if (root.hasChildNodes)
        list(1,0,0);                              //默认打开第 1 章的内容
}

function list(zh,jie,wt){
//以下取得当前要操作访问的结点
currentnode=root.childNodes.item(zh-1);
if (jie!=0){
   currentnode= currentnode.childNodes.item(jie-1);
   if (wt!=0)
      currentnode= currentnode.childNodes.item(wt-1);
}
url=currentnode.getAttribute("url");              //得到当前结点的 URL 地址
if (url!="") {
     top.main.location=kcpath+"/"+url;            //在 main 帧中显示 URL 内容
}
/* 以下拼接导航菜单显示的文本串 */
str="<table border=0 cellPadding=0 cellSpacing=0><td width=20><td width=20><td width=20><td
width=20><td width=90%    align=left><tr>";
m=root.childNodes.length;                         //求章数
for (i=1;i<=m;i++) {                              //处理所有章
x=root.childNodes.item(i-1).getAttribute("title");
str+= "<td><img border=0 src='midplus.gif' ></td>"
```

```
str+= "<td><img border=0 src='close.gif' ></td>"
str+= "<td onclick='return list("+i+",0,0)' style='cursor:hand' colspan=3   "
str+= "align=left><nobr>"+"<font color='yellow'>"+x
str+= "</font></nobr></td><tr>";
if(zh!=i) continue                                  //如果不是当前章，则跳过以下代码
n = root.childNodes.item(i-1).childNodes.length;    //求本章节数
thiszhang= root.childNodes.item(i-1);               //得到当前章结点
for (p=1;p<=n;p++) {                                //当前章要展开相应节，处理本章所有节
x2= thiszhang.childNodes.item(p-1).getAttribute("title");
str+= "<td ><img border=0 src='line.gif' ></td>"
str+= "<td><img border=0 src='midblk.gif' ></td>"
str+= "<td><img border=0 src='close.gif' ></td>"
str+= "<td colspan=2 align=left onclick='list("+i+","+p+",0)'"
str+= " style='cursor:hand'><nobr>"+x2+"</nobr></td><tr>";
if(jie!=p) continue                                 //如果不是当前节，则跳过以下代码
n2= thiszhang.childNodes.item(p-1).childNodes.length;  //当前节的问题数
thisjie= thiszhang.childNodes.item(p-1);            //当前节结点
for (t=1;t<=n2;t++) {                               //展开当前节的所有问题
t1=t-1;
str+= "<td><img border=0 src='line.gif' ></td>"
str+= "<td><img border=0 src='line.gif' ></td>"
str+= "<td><img border=0 src='midblk.gif' ></td>"
str+= "<td width=20><img border=0 src='close2.gif'></td>"
str+= "<td   align=left   onclick='list("+i+","+p+","+t+")' "
str+= "style='cursor:hand'><nobr><font color='99FFCC'>"
str+= thisjie.childNodes.item(t1).getAttribute("title")
str+= "</font></nobr></td><tr>";
 }                                                  //问题循环
}                                                   //节循环
}                                                   //章循环
str=str+"</table>";
menu.innerHTML=str;
}
</script>
</head>
<BODY onload="init()"   background="images/nav.gif" ><center>
<div id="menu"> </div>
</center></body></html>
```

【说明】注意 list 函数的 3 个参数与当前访问的章节知识点的关系，程序中用了三重循环来展开当前访问的知识点，只有访问到问题时才会实际执行到第三重循环。当前知识点的 URL 内容的课程路径信息（kcpath）存储在 Cookie 变量中，在用户进入课程时通过查阅数据库中课程的内容路径信息给 Cookie 赋值，每门课程的内容放在一个独立的路径下。为了实现内容的逐级缩进显示，用到几个显示目录树的小图片，并通过表格进行定位，此时需注意表格单元格的宽度和数量设置，图片所在单元格的宽度基本上就是图片的大小，这样可形成紧凑的内容排列。

11.4　网络考试系统

网络考试系统是网上教学系统的一个较为复杂的部分，涉及的环节比较多，包括考试的进入控制、考试的组卷、考试试题及解答界面的显示设计、考试的限时、考试成绩的计算、学生解答的记录、考试分析等。以下就学生端考试功能的主要部分进行重点介绍。

11.4.1　考试界面布局

整个考试界面由 3 帧组成，如图 11-9 所示。上面的一帧用来进行定时显示控制，将显示剩余时间，并在剩余时间用完时自动提交答题卡的交卷表单；左边一帧显示试题内容；右边一帧显示答题卡。整个框架页面代码如下：

```
-------------------------------------------------- examlist.asp -----------------------------------------------------------
<frameset border="0" framespacing="0" rows="27,*" frameborder="0">
      <frame name="up"   src="examtime_limit.asp" scrolling="no">
      <frameset border="1" framespacing="0" cols="72%,*">
         <frame name="menu"   src="paper_list.asp" scrolling="auto" >
         <frame name="right"   src=" answer_card.asp" scrolling="auto" >
      </frameset>
</frameset>
```



图 11-9　考试界面

11.4.2　考试组卷程序

学生若想进入考试环节，首先要根据组卷策略产生试卷。系统中包括 5 类试题，各类试题的抽题策略是一致的，只是从不同数据库表格中抽题。因此，系统中专门设计了一个函数 pickst 实现抽题功能，其中包括数据库连接、表格、抽题数量、考核知识点范围、难度共 5 个参数。

1．从题库中根据组卷条件选取试题的函数

不同类型的试题存储在不同的表格中，所有这些表格的字段类似，其主要字段说明如下（注：本例只用到前 3 个字段）。

- st_no：试题编号。
- knowledges_point：试题所属知识点，本系统中每道试题只能属于某个知识点。
- difficulty：试题难度系数。
- content：试题内容。
- answer：标准答案。

以下为一个包含文件，其中给出了选题函数的定义，具体代码如下：

```
--------------------------------------------------- pickupst.inc ------------------------------------------------------------
<script language="vbscript" runat=Server>
function pickst(conn,   table , count1 ,knowledges,diff)
  'conn 为数据库连接，table 为数据库表格，count1 为选题数量
  'knowledges 为考核知识点集合，diff 为难度
if count1 = 0 Then
  pickst = ""                                    '选题数量为 0，函数返回空串
else
  Randomize
  point = split(knowledges, ",")                 '分离出考试知识点
  amount= Ubound(point)-1                        '求考核知识点数量
  havepick =""                                   '用于拼接选好的试题的编号，中间用逗号隔开
  set rs = Server.CreateObject("ADODB.recordset")
  for xx = 0 To amount                           '循环在每个知识点选题
    sql = "select * from   " + table + " where   difficulty='" + CStr(diff) +
            "' and knowledges_point like '" & point(xx) & " '"
    rs.Open sql, conn, 1, 1

    '以下程序段求每个知识点预选试题数量并存入 precount 变量中
    if rs.EOF then
      precount = 0                               '无试题可选，则预选试题数量置为 0
    else
      px = rs.RecordCount                        '知识点试题总数
      precount = Int(count1 / amount + 0.5)      '均分每个知识点预选试题数量
      if px < precount Then
          precount = px
    end if
```

```
        end if

        shiti = split(genrandom(precount, px) , ",")
         '调用例 3-6 的函数 genrandom 产生随机数列并存入数组 shiti 中

        '以下根据产生的随机数列作为记录序号从数据库表格记录读取试题编号
        rs.movefirst
        start=rs.bookmark
        for   kk =0 to precount-1
            rs.Move shiti(kk)，start                     '移动到随机数列元素指定记录
            n = rs.Fields.Item("st_no")                '读取数据库中试题编号字段
            havepick = havepick + CStr(n) + ","
        next
        rs.Close
    next
    pickst = havepick
end if
end function
</script>
```

【说明】本程序在各个知识点上均分进行选题，首先利用第 3 章编写产生随机数列的函数产生记录编号，然后读取相应记录的试题编号。将所有选取试题的编号拼接为一个字符串作为返回结果。由于选题函数在多处使用，所以将其安排在一个 inc 文件中，使用时用 include 语句将其包含到具体的 ASP 程序中。

2．组卷 ASP 程序

组卷 ASP 程序涉及的 exam_set 表格为组卷设置记录表，包括如下字段。

（1）kcname：课程标识。

（2）exam_point：考核知识点。

（3）diff：难度系数。

（4）danxuan_amount：单选题数量。

（5）danxuan_score：单选题每道题的分值。

（6）……其他字段，如多选、是非、填空等均涉及试题数量和分数。

以下为程序的主要代码：

```
<!-- #include file = "pickupst.inc" -->
<%
file= "database.mdb"
Set Conn=Server.CreateObject("ADODB.connection")
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Conn.Open connstr
'取得课程的考试要求
sql1="Select * from exam_set where kcname='"&request.cookies("kcname")&"'"
set rs1=conn.execute(sql1)
xzt_amount = rs1("danxuan_amount")
diff = rs1("diff")
```

```
knowledges = rs1("exam_point")
response.cookies("exam")("danxuan") = pickst(conn,"danxuan_table",
        xzt_amount,knowledges,diff)  '调用选题函数挑选试题
…… 其他类试题的选题略 ……
response.redirect("examlist.asp")     '转向考试内容显示页面
%>
```

【说明】该程序根据考试设置中各类试题的选题要求（包括选题数量、难度、知识点范围等）调用选题函数，从相应的试题库表格中选题，并将选好的试题编号保存到 Cookie 变量中，以便在后续页面中能够进行处理。

11.4.3　试题显示处理程序

试题显示处理程序的功能是根据 Cookie 变量中记录的选题访问数据库，然后将试题内容显示在框架页面中。

```
---------------------------------------------- paper_list.asp ---------------------------------------------------------
<html><head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style>
.body {font:14pt  宋体}
.table {font:14pt; color:"black"}
</style>
</head>
<body >
<table width=99% style="table-layout:fixed">
<td style='word-wrap:break-word'>
<%
dim shiti
file= "database.mdb"
Set Conn=Server.CreateObject("ADODB.connection")
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Conn.Open connstr
result = request.cookies("exam")("danxuan")
shiti = split(result,",")
p=Ubound(shiti)-1
%>
<font color="red">一、单项选择题</font>
<br>
<% i=0
do while i<=p                                          '循环显示各道试题的内容
sql="SELECT * from jvselect where st_no="&shiti(i)         '查询试题
set rs=Conn.execute(sql)
if not rs.eof then
 %>
<pre><font color="blue">
<b><%=(i+1)%>.</b></font><%=server.htmlencode(rs("content"))%>
```

```
</pre>
<%
end if
i=i+1
loop%>
…… 其他试题显示类似，这里省略 ……
</body>
</html>
```

【说明】根据所选试题编号访问数据库，并显示试题内容，显示时要注意两点，一是用 Server 对象的 htmlencode 方法对试题内容进行转换，以便试题内容显示保持原来的面貌；二是采用表格设置固定宽度，在超出宽度时让内容自动折行，这一点是通过表格和单元格的样式设置实现的。

11.4.4　答题卡显示处理程序

答题卡显示处理程序的功能是根据 Cookie 变量中记录的选题数量，生成答题卡的解答控件供用户解答。

```
---------------------------------------------- answer_card.asp -----------------------------------------------------
<html>
<head>
<STYLE type=text/css>
body {      COLOR:blue; FONT-FAMILY: "宋体"; FONT-SIZE: 12pt }
TABLE {   COLOR:black; FONT-FAMILY: "宋体"; FONT-SIZE: 10pt }
</STYLE>
</head>
<body>
<center>
<big>答题卡</big>
<form name="my" method="post" target="_top" action="exam_remark.asp ">
<%
dim shiti                                                    '用来存放试题的数组
result = request.cookies("exam")("danxuan")             '获取选好的试题编号序列
shiti= split(result,",")                                     '将试题编号存入数组 shiti 中
%>
<font color="red">一、单项选择题</font>
<br>
<table width="100%" border="0">
<%
i=0
p = ubound(shiti)-1
do while i<=p                                                '循环产生各道试题的选项
 %>
<tr>
<td align="left"><font color="blue"><b><%=(i+1)%>.</b></font>
<%                                                           '以下生成一道试题的 5 个选项的解答控件
```

```
for t=1 to 5
  id = mid("ABCDE",t,1)                    '选项的标识
  name="sx"&i                              '解答选项的名称，以 sx 作为前缀
%>
<td align="left">
<input type="radio" name=<%=name%> value=<%= id %>><% = id %></font>
</input>
</td>
<%   next %>
</tr>
<%
i=i+1                                      '下一道题
loop %>
</table>
…… 其他类试题的答题卡显示处理类似，此处略……
<br>
<input type="submit" name="my" value="   交   卷   " > </input>
</form></center>
</body>
</html>
```

【说明】答题卡的显示与试题内容无关，而只与试题数量产生和每道试题对应的解答控件有关。要给每个试题进行编号，解答控件的命名很重要，在后续页面中将根据解答控件读取对应试题的学生解答。单选题控件的命名采用前缀 sx 加试题序号的方法。

11.4.5　交卷评分显示处理程序

交卷评分显示处理程序将用户的解答与标准答案进行比较，并根据数据库中保存的每小题分值计算得分。另外，该程序要做的工作还包括记录学生的得分以及记录学生考卷、解答情况等信息，为了节省篇幅、突出主线，这部分内容在程序中省略。需要注意的是，以下代码中仅包括单选题的评分处理部分，而其他试题的评分处理部分省略了。

```
--------------------------------------------- exam_remark.asp ----------------------------------------------------------
<%
score=0
file= "database.mdb"
Set Conn=Server.CreateObject("ADODB.connection")
connstr="driver={Microsoft Access Driver (*.mdb)};dbq="&Server.MapPath(file)
Conn.Open connstr
'取得课程的考试评分
Sql ="Select * from exam_set where kcname='"&request.cookies("kcname")&"'"
set rs1=conn.execute(sql)
score1=rs1("danxuan_score")                '单选题的每小题分值
……其他类试题的每小题分值省略 ……
%>
<html>
<head>
```

```
<script language="javascript">
  function check (x){
    if (x>=90) {
        alert(" 你的得分： "+x+"\n 祝贺取得优异成绩!! "); }
    else if (x<90 && x>=80)
        {alert(" 你的得分： "+x+"\n 考试成绩不错! "); }
    else if (x<80 && x>=70)
        {alert(" 你的得分： "+x+"\n 考试成绩尚可! "); }
  else if (x<70 && x>=60)
        {alert(" 你的得分： "+x+"\n 考试成绩一般，\n\n 还需要不断进步!"); }
  else if (x<60)
        {alert(" 你的得分： "+x+"\n 希望你加倍努力学习 ！"); }
    window.location = "student_do.asp ";    //返回学生操作主界面
  }
</script>
</head>
<body >
<%   '以下首先对单选题进行评分处理
dx=request.cookies("exam")("danxuan")          '考试用到的单选题编号序列
shiti=Split(dx,",")                            '将单选题编号存入数组中
amount = Ubound(shiti)-1
i=0
do while i<=amount
     sql="SELECT * from jvselect where st_no="&shiti(i)
     set rs=Conn.execute(sql)
     x = rs("answer")                          '获取标准答案
     rs.close
     y = request.form("sx"&i)                  '读取用户解答
     if x=y then
          score=score+score1                   '统计得分
     end if
     i=i+1
loop
……考试成绩及试卷登记的代码略……
<script language="javascript">
    check(score);     //提示用户得分
</script>
</body></html>
```

【说明】评分结果通知学生是以弹出对话框的方式进行的，之后将页面的 URL 重定向到学生操作的主页面。如此，可防止学生用浏览器的“回退”按钮返回到测试界面。

本 章 小 结

本章选择网上教学中的 4 个典型应用进行介绍，这 4 个应用代表了不同的特色。单元测试是在浏览器客户端访问 XML 试卷实现动态交互处理的应用；导航菜单的应用则代表

了客户端技术与服务器端技术的结合，综合利用了客户端技术和服务器端技术改进应用效率。最后介绍的考试系统虽然只是实际系统的简化描述，但从中可发现各类技术的综合应用。

习　　题

1．参照网上答疑系统设计实现网上作业系统的功能。要求教师可布置作业，学生可解答作业，教师可批改作业，学生可查看教师的批改意见。

2．设计一个网上家教园地，分为家庭教师账户和学生账户，包括首页、家教注册模块、学生注册模块、家教信息浏览模块、学生信息浏览模块、家教个人信息管理模块、学生个人信息管理模块等。将模块功能划分到账户。

3．设计一个新闻发布系统，分为管理员和普通用户两类，管理员可录入、删除、修改新闻，而普通用户可查看新闻。查看新闻先列新闻标题、点击标题列出新闻详细内容，热点新闻和最近新闻排列在前面显示。点击次数多的为热点新闻，另外，支持分页显示和模糊查询新闻的功能，用户还可对新闻进行投票。

【提示】数据库表格包括 3 个表，第一个是用户信息表，可包括登录名、密码、角色、姓名等字段；第二个是新闻表，可包括新闻标题、编号、新闻内容、发表时间、点击次数等字段；第三个表是新闻投票记录表，包括新闻编号、投票等级、投票者等字段。

4．设计一个简单网络考试系统。分为教师、学生两类用户。

教师操作部分的功能包括如下方面：

（1）可管理试题库中的试题，假设试题库中只包括单选题。

（2）考试安排，如抽题数量、考试时间限制等。

（3）查看所有学生考试成绩。

（4）清除本次考试成绩。

学生操作部分的功能包括如下方面：

（1）参加考试，系统随机抽题，考试完毕后系统记录学生成绩。

（2）查看所有学生的考试成绩。

5．以下代码是网络教学系统中实现考试限时处理代码的简化版，仔细阅读代码的实现并为书上介绍的单元自测应用增加限时功能，到达时间后自动交卷。

```
<SCRIPT language="JavaScript">
var limit;                          //考试限时值（单位为分）
var now;                            //现在时间
var t1;                             //记录考试开始时间，折算为秒
function init(){
   limit=100;                       //假设考试限时 100 分
   now=new Date();
   t1=now.getHours()*3600+now.getMinutes()*60+now.getSeconds();
   computetime();
}
function computetime(){             //计算并显示剩余时间，时间用完后自动交卷
```

```
  now=new Date();
  b2=now.getHours();
  m2=now.getMinutes();
  s2=now.getSeconds();
  passtime=b2*3600+m2*60+s2-t1;                    //计算当前时间与考试开始时间差
  remain=limit*60-passtime;                        //计算剩余时间（秒）
  remainhour=Math.floor(remain/3600);
  remainminute=Math.floor((remain-remainhour*3600)/60);
  remainsec=Math.floor(remain-remainhour*3600-remainminute*60);
  if (remain<=0) {
      parent.frames("right").document.my.submit();   //自动交卷
   }
  else{
    str="考试剩余时间：<font color=red>"+remainhour+"</font>小时: "
    str=str+"<font color=red>"+remainminute+"</font>分: "
    str=str+<font color=red>"+remainsec+"</font>秒";
    document.all.me.innerHTML=str;
    setTimeout("computetime()",1000);
  }
}
</SCRIPT>
<body onload="init()">
<div id="me"></div>
</body>
```

实际应用的运行结果如图 11-9 所示。

【说明】此时的剩余时间显示缺少很多防范措施，如学生修改系统时间，学生单击刷新按钮等均将导致计时不准，如何修改程序防范这些问题？

【提示】防范学生修改时间就要设法记住时间，防止学生刷新就要设法将剩余时间记住。利用 Cookie 是一个办法，上网查阅如何在客户方用 JavaScript 读写 Cookie。

参 考 文 献

1．丁振凡．Web 程序设计．北京：北京邮电大学出版社，2008
2．潘晓南．动态网页设计基础（第二版）．北京：中国铁道出版社，2008
3．张景峰．ASP 程序设计教程（第二版）．北京：中国水利水电出版社，2007
4．林弘之．Web Services 原理与开发实务．北京：电子工业出版社，2003
5．吉根林．Web 程序设计（第 2 版）．北京：电子工业出版社，2006
6．（美）Steven Holener.XML 完全探索．北京：中国青年出版社．师夷工作室译，2001
7．丁振凡．XML 数据岛技术及应用，微型机与应用，2002
8．丁振凡．XML 技术在网络课件导航菜单改进中的应用，福建电脑，2006
9．丁振凡．AJAX 技术在网络考试系统分析中的应用，华东交通大学学报，2007
10．丁振凡．利用 XML 实现网络课件中的习题操练，计算机系统应用，2002
11．丁振凡．基于 XML 的单元自测应用的设计，华东交通大学学报，2008